A Component Assembly Approach to Support Human-computer Interface Construction

Qingtao Wu
Electronic and Information Engineering College, Henan University of Science and Technology, Luoyang , China
Email: wqt8921@126.com

Mingchuan Zhang, Ruijuan Zheng and Wangyang Wei
Electronic and Information Engineering College, Henan University of Science and Technology, Luoyang , China
Email: {zhlzmc, rjwo, weiwangyang}@163.com

Abstract—**A good design of human-computer interface can make the communication more effective, more easily and less mistaking guidance for users. In order to realize the rapid construction and dynamic reconfiguration of human-computer interface, a rapid developing model based on component assembly was proposed recurring to an assistant design tool developed self-owned, as well as the assembly models of domain component and interface components. In the rapid construction model, the scheduling component cooperated with configure file to achieve the established functions. In the meantime, the interface switching were realized by event handler, rule handler, interface forming handler and addressing handler through cooperating with configure file that was acquired by assistant design tool. Finally, the example of system realization showed that the rapid construction method was feasible.**

Index Terms— **Distributed System, Human-computer Inerface, Domain Component, Interface Component, Component Assembly**

I. INTRODUCTION

With the development of computer technology, users have higher expectations for computer system, especially to the human-computer interactive system. Not only do users require that the human-computer interactive interfaces(HCII) are beautiful, easy to use, response sensitive, but also it can be constructed rapidly and configured dynamically.

Recent years, the dynamic reconfiguration problems are paid attention to by more and more researchers in distributed system. Remote dynamic component configuration is discussed, which greatly improves system flexibility using configuration files[1]. The dynamic deployment and re-configuration of pervasive service components in a self-controlled manner are researched. In particular, a service component self-deployment algorithm using partitioning techniques and a simple service re-configuration algorithm are proposed and evaluated. The effectiveness of the proposed mechanisms is proved by the experiment results [2].

A new method of QoS-aware and dynamic configuration for Web services composition is presented to improve the adaptive capacity to both the QoS variability of component services and the failure-prone environment [3]. There are two topics which are researched in reference [4]. First, it describes optimizations applied to an implementation of the OMG's Deployment and Configuration of Components specification that enable performance trade-offs between QoS aspects of DRE systems. Second, it compares the performance of several dynamic and static configuration mechanisms to help guide the selection of suitable configuration mechanisms based on specific DRE system requirements. Two methods of dynamic reconfiguration are introduced. First, using configuration file, this method belongs to static configuration ways. Second, utilizing configuration operation in the program, this method belongs to dynamic reconfiguration ways, which can adapt to some configuration situations that can't be estimated in advance. The software architecture supporting dynamic reconfiguration is studied in reference [5]. It is solved with the graph-oriented programming method, which realizes dynamic reconfiguration and the description of software architecture based on components in the uniform way. Certain problems of components dynamic reconfiguration are researched in references [3-5], and some achievements are got. However, there are some difficulties in practical application. Furthermore, systemic fault-tolerant has not been considered. Component frameworks provide the strategy for the development and deployment of complex multiphysics applications to satisfy the need [6]. In order to build dynamically adaptable applications, the service-oriented component models supporting the dynamic availability of components at run-time are researched as well as offering the possibility in reference [7]. A new component replacement analysis method to solve asset replacement problems for complex electricity distribution systems is developed[8].

By using the Plug in tool of the creator and language of Open Flight API in the Visual C++ compiler environment, components could be added into. The component modeling method is applied to engineering project to acquire the certain application value[9]. The predictable component technology and the COMTEK-λ technology are analyzed and an example of PECT is given, which used CIMTEK technology and end-to-end delay of the delay predication model[10]. In the process of the development of a federation complied with Federation Development and Execution Process, there is a new pattern of the development of the federation, which is developed not on the base of a federate, but on the base of simulation model component. The technique of the composing of simulation model components is important to construct simulation system[11].

The human factors in the design of process control systems are argued and two types of human-machine interfaces are distinguished[12]. Reference [13] deals with a screen-based ecological interface designed to support the operators' work in a medium voltage electricity distribution control room. More specifically, the focus is placed on how the designed interface supports operators in skill-, rule-, or knowledge-based behavior, as well as in the transition between the three levels of behavior. However, the construction method of human-computer interface based on component assembly is researched scarcely because of the complexity and variability.

This paper considers rapid construction and dynamic reconfiguration for HCII. A series of methods are presented to realize the rapid construction for HCII based on domain component and its assembly.

II. COMPONENT ASSEMBLYAND DOMAIN COMPONENT

A. *The concept of component assembly*

The component assembly is one of the key technologies in the field of software component. It can be expressed as in formula (1)[14].

application = components + composition language (1)

Over here, composition language(CL) can be expressed as in formula (2).

CL = composition operators + glue logic (2)

The component assembly is a process that adopts certain composition language to realize glue logic and assembles components into a system or advanced components according to composition operators. The components that are used to assembly are called elementary components. The component produced is called composite component. The complex component produced is a software system.

B. *Domain component*

1) Domain component

The domain component is the component that is restricted by special domain. That is to say, the domain component is the component that applies in the special domain. The domain component has further restriction than ordinary component, but it has further certainty, which is exhibited as follows.

- Component granularity: it is determined by the Property of correlative business logic in the special domain.
- Component environment: it is more simplex and steady than the component environment of ordinary component.
- Business logic: it is more canonical and standard than the business logic of ordinary component.

2) Domain application framework

For a special domain, system requirements have consequentially the commonness which is described by domain model. Domain specific software architecture (DSSA) proposes the resolvent aiming to the domain model. DSSA is a high level architecture that adapts the various system requirements in special domain. It includes components and their assembly rules. The components are divided into business components and framework components. Domain application framework(DAF) that is an instantiation of DSSA images the architecture of software system family and provides the basic construction mode to establish the software architecture.

The assembly process that the application is constructed through assembling DAF and business components is shown as in figure 1. When a certain system is developed, the DAF is realized by framework component which is simply called framework to differ from business component. The DAF based on component has three properties which are shown as follows.

- The DAF aims at the special domain.
- The DAF is an instantiation of DSSA.
- The DAF consists of a group of concurrent components.

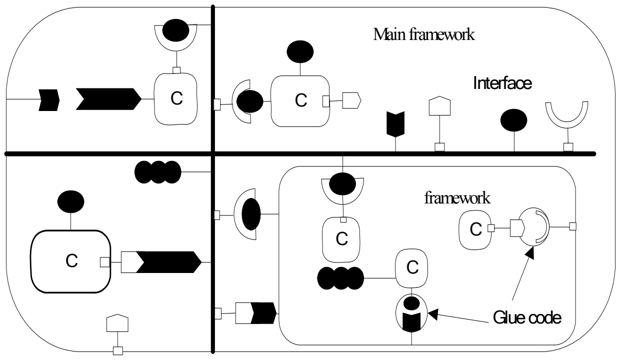

Figure 1. the model of component assembly

III. ASSEMBLY MODEL OF INTERFACE COMPONENT

Not all interface components can be assembled directly to form a target system. The components that can be assembled should satisfy the special restriction. Moreover, the software development process that utilizes component assembly to construct target system differs from the ordinary software development process[15].

A. *The criterion of interface component assembly*

Generally, the operator of component assembly can be divided into two stages. One is to design, develop and describe the components that possess assembled property. The other is to assemble components according to actual requirement. In order to realize assembly, the criterions should be followed.

- The frameworks need to match with runtime environment, while components need to match with frameworks.
- If the frameworks do not match the runtime environment, the matching can be achieved by glue codes which are used to framework.
- If the main frameworks match with main frameworks, the interfaces and platform need to be matched.
- When establishing criterion of component assembly for special domain, enumerating method is supposed to act as the preferred method.

B. *The assembly modle of interface component*

The assembly process of interface component is a process that the assembly system selects components. The core operator is the matching of frameworks and frameworks (or business components). The assembly system selects a framework acting as the main framework according to the special requirement. The main framework selects one or more frameworks or business components according to business logic. If business components are selected, the matching process finishes for this matching branch. If frameworks are selected, the matching process continues until all branches match business components.

There is at least a framework that is called main framework for one application. The main framework denotes the business logic relation of the application. The business components and frameworks are assembled by the main framework to realize the application business logic. The assembly model of components and frameworks is shown as Figure 2. Because the main framework is the first entity assembled and the function of other components is to cooperate with the main framework, its actual function is a component vector. The conclusions can be achieved from figure 2.

- Business components can only be assembled to frameworks, but can not be assembled to business components.
- Frameworks can be assembled to other frameworks. That is to say, simple business logic combines with other business logic to gain complicated business logic.
- The workflow of application is determined by the frameworks. The main framework calls frameworks and business components. The higher frameworks call lower framework and business components. The system function is realized by iterative. The calling flow can be reversed.
- The converting, mapping and gulling for in-matching interfaces are done by frameworks.
- An application includes at least a main framework that realizes special business logic through assembling sub-framework and business components.
- The main framework is the first component assembled. It expresses the runtime environment and is the vectors of all business components actually.

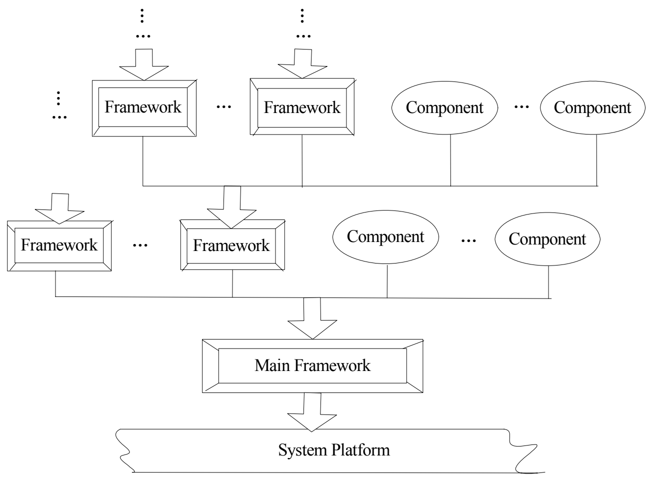

Figure 2. The assembly modle of interface component

IV. THE RAPID DEVELOPING MODEL OF HCII

A. *The design flow of HCII*

Generally, the traditional design flow of HCII is divided into requirement analysis, system analysis, interface design analysis, interface style selecting, interface portions dividing, interface portions implementing and system testing. In the design process, the interface design criterion that is needed by the customers or target system should be consulted. The design flow of HCII based on domain component assembly has some difference. It is divided into requirement analysis, system analysis, interface component dividing, interface component selecting and system testing. If the interface component selected is nonexistent, it needs to be developed. The traditional design flow and the design flow based on domain component assembly are shown as in figure 3.

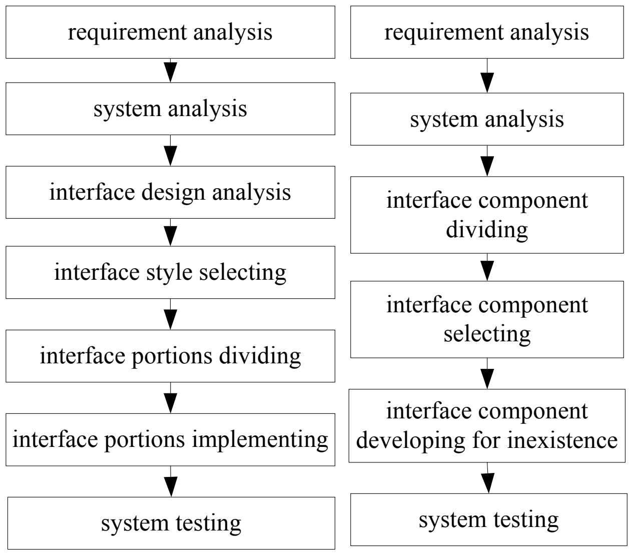
requirement analysis
system analysis
interface design analysis
interface style selecting
interface portions dividing
interface portions implementing
system testing
requirement analysis
system analysis
interface component dividing
interface component selecting
interface component developing for inexistence
system testing

Figure 3. The design flow of HCII

B. Assistant design tool

In the design process of HCII based on domain component, the components need to be made certain to construct target system after requirement analysis. However, the construction process that utilizes interface components to construct target system is very complex. Therefore, an assistant design tool is developed independently to assist the interface design of HCII. The functions and operation methods are introduced as follows.

1) The functions of assistant design tool

- Interface component query: the queries results include the description and display snapshot(vectorgraph or photograph) of inter face components.
- The snapshot design for new interface components: if an interface component which is absent is selected, its simple snapshot should be designed by assistant design tool to continue the design process of HCII.
- Interface organizing: the final interface style is organized through assembling organically all snapshots of interface components selected.
- Dynamic displaying of final interface: the final interface style is displayed to be made certain by designer or customers.
- Saving design results: the design results are saved to provide to programming.

2) The operation methods of assistant design tool

Two operation methods are introduced in this section. One is to put snapshots of interface component into snapshots library. The other is to assist designer to develop HCII. The operation flows are shown as in figure 4 and figure 5.

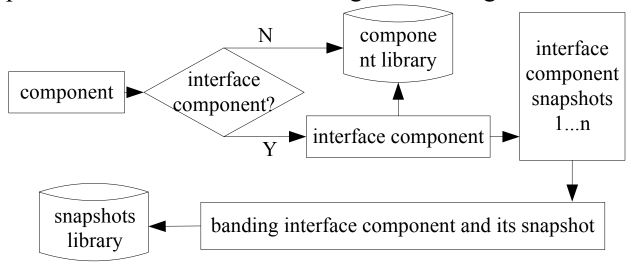

Figure 4. The operation flow of putting the snapshot into library

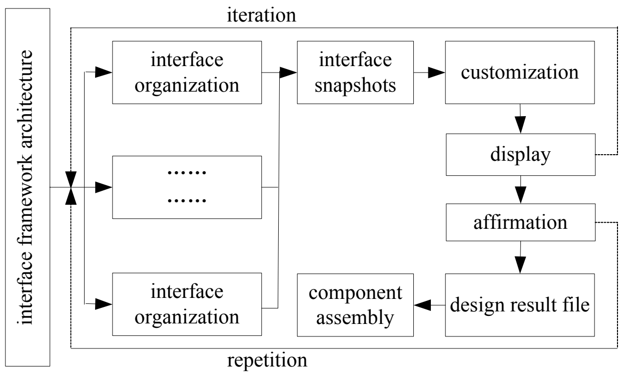

Figure 5. The operation flow of assisting deisgner

C. The developing model of HCII

The human-computer interactive system can be developed rapidly recurring to the assistant design tool. Each component in the component library has its snapshot which can be used directly to build the interface snapshot of the target system. For the components nonexistent in component library, the interface snapshot can be designed by the assistant design tool.

When designing a HCII, the snapshot of human-computer interface is gained recurring to the assistant design tool, as well as the design result file. The target system can be build based on the file. The developing model is shown as Figure 6 and the developing steps are shown as follows.

- Determining components: abstracting the requirement and determining system framework model to gain the components which build the target system based on the requirement analysis.
- Determining the flow of interface switching: analyzing application workflow to gain the flow of interface switching.
- Gain designing result: generating interface designing result file which includes the ID of framework, the ID of business component, the ID of the component needing developing and interface switching information based on the step one and step two.
- Component developing: developing the new components whose ID is generated in step three and putting them into the component library.
- Component assembly: assembling the various components to gain target system according to the design result file which is generated in step three.
- Assembly testing: testing the target system, if the target system satisfies the need, the designing process is accomplished. If not, the process will be iterative.

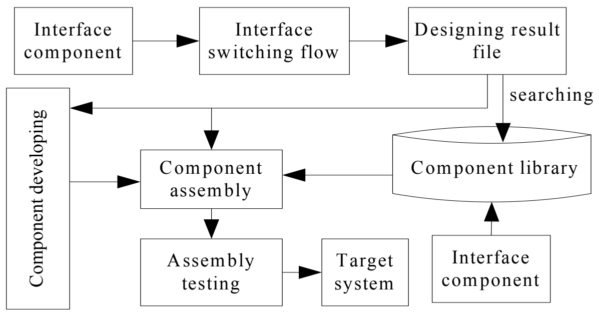

Figure 6. the rapid developing model of HCIS

V. THE CONSTRUCTION OF DISTRIBUTED HCII

A. System deployment

The system deployment is done by deployment programmer. When constructing a distributed human-computer interface, the deployment programmer deploys the interface components, scheduling components, frameworks and configure file for each node according to the design results files which are generated by assistant design tool. Each node has a scheduling component, a configure file, several frameworks and several interface components. The scheduling component and configure file are the core of each node. The scheduling component realizes the business logic through scheduling frameworks and interface components according to configure file. The deployment model is shown as in figure 7.

1) Scheduling component

The interface scheduling process is illustrated using example of node 1 as follows. According to requirements, the operator of node 1 does an action which is called an event. This event affects possibly the interior or exterior of component generating event. If the effect is inner, the component generating event deals with the event by self. If the effect is outer, the scheduling component forms a new frame interface according to the rules which is generated by assistant design tool and described in configure file.

2) Configure file

The configure file that determines the interface switching is generated by assistant design tool. It includes the event id, interface form, component id, component memory address, framework id, framework memory address, interface switching flow, etc.

3) Frameworks

The frameworks refer to all the frameworks deployed in the certain node.

4) Interface components

The interface components refer to all the interface components deployed in the certain node.

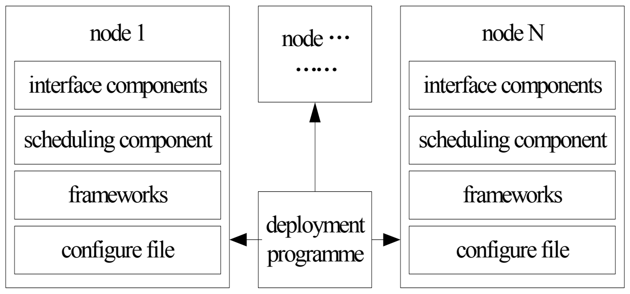

Figure 7. The deployment model of HCII

B. System running

When the system is running, each node works normally according to the user's operation and the switching flow determined by configure file. The interface switching is taken charge by scheduling component which includes event handler, rule handler, interface forming handler and addressing handler. The flow chart of scheduling component is shown as in figure 8.

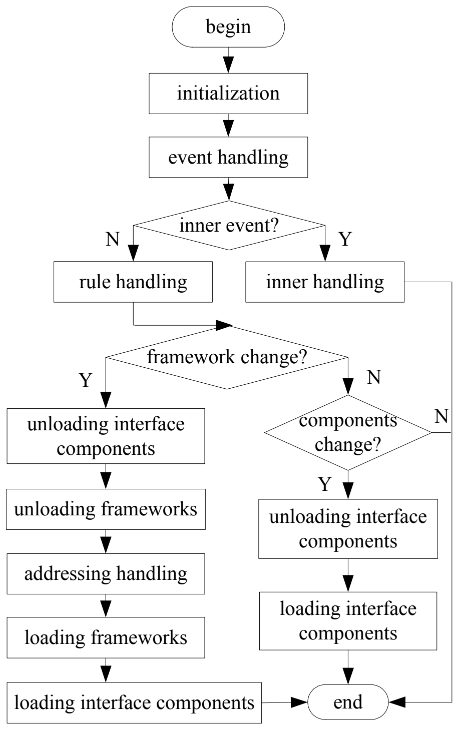

Figure 8. The flow chart of scheduling component

1) Event handler

The event handler monitors whether there are events produced or not. If there are events produced, the event whose PRI is highest is resolved according to event library that is record in configure file. The results resolved such as event id, corresponding rule id are sent to rule handler.

2) Rule handler

When receiving a rule id, the rule handler resolves the rule to generate components lists and their assembly relationship according to rule library which is record in configure file.

3) Interface forming handler

The interface forming handler forms the display interface according to the components lists and their assembly relationship that are generated by rule handler.

4) Addressing handler

When forming the final display interface, the memory addresses of components are made sure by addressing handler to load components.

VI. An Example of System Realization

The architecture of example is made up of Information Process System, Application System, Data-Command Agency and Human-computer Interactive System. The software platform is CORBA. Among them, the Information Process System produces and processes the simulation necessary data. The Data-Command Agency manages the data and command uniformly. The architecture is shown as in figure 9.

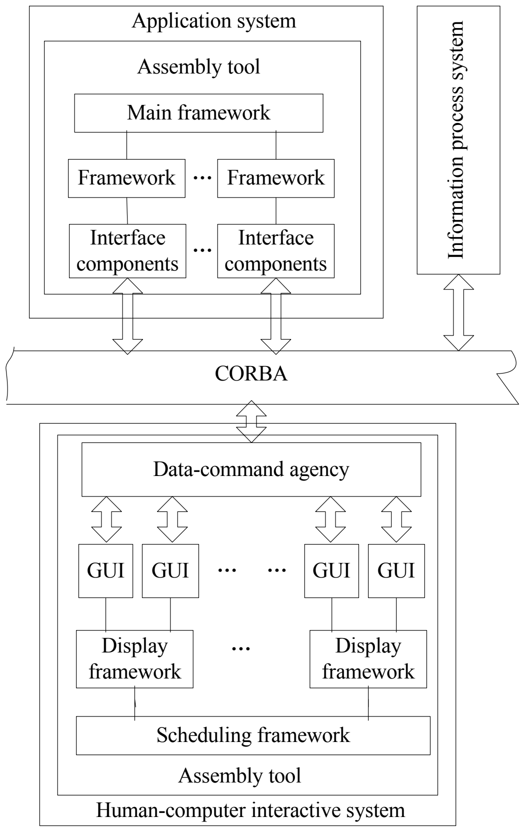

Figure 9. the architecture of example

The scheduling model of the Human-computer Interactive System is determined according to the configure file which is generated recurring to the assistant design tool. In the model, the nodes cooperate with other nodes to accomplish the scheduling business logic. The event handler receives events, processing them and sending them to rule handler according to the event library. The rule handler receives and processes rules to generate interface forming information. When scheduling happens, the memory addresses of components are made sure by addressing handler to load components and accomplish interface switching. The scheduling model is shown as Figure10.

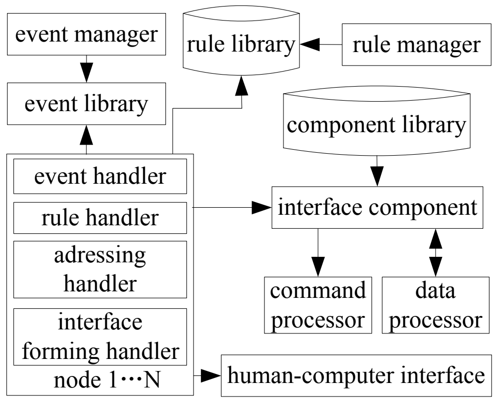

Figure 10. the scheduling model

VII. Conclusion

Three achievements are acquired in this paper. First, the assembly models of domain component and interface component are presented which is the base for further research. Second, the rapid construction model and method of distributed human-computer interface are proposed recurring to an assistant design tool developed self-owned, as well as the functions and operation methods of the tool are introduced. Third, the feasibility of the rapid construction model is proved by the example of system realization.

There are still some problems remained to study such as the granularity division of the interface components, the mathematic model of effect evaluating of scheduling and composition of interface components, which will be researched in the future work .

Acknowledgment

The financial support from the National Natural Science Foundation of China (NSFC) under grant No. 61003035 and Henan Natural Science Foundation of China under grant No. 072300410210 is acknowledged gratefully.

References

[1] Lu, Liu; Zongyong, Li; Ruibo, Li. “Improving information system flexibility through remote dynamic component configuration”. International Conference on Service Systems and Service Management(ICSSSM 06). Oct.2006, pp. 461-466.

[2] Ou, Shumao; Liu, Dongsheng; Yang, Kun. “Dynamic algorithms for self-deployment and self-configuration of pervasive service components”. Intelligent and Software Intensive Systems (CISIS 09) IEEE Press, Mar. 2009, pp. 525-530.

[3] Yang, Huaizhou, Li Zengzhi. “Research on QoS-aware and dynamic configuration of web services composition system,” Journal of Xi'an Jiaotong University, vol. 44, Feb.2010, pp.25-30.

[4] Venkita Subramonian, Gan Deng, Christopher Gill. “The design and performance of component middleware for QoS-enabled deployment and configuration of DRE systems,” Journal of Systems and Software, Vol.80, May 2007, pp.668-677.

[5] Wu Haomin, Cao Min. “Description of Software Architecture Supporting Dynamic Reconfiguration and Abstract Programming”. Computer Engineering & Applications. Oct 2004, pp.94-98.

[6] Brandt Steven, Allen Gabrielle, Eastman Matthew. “Dynamic deployment of a component framework with the Ubiqis system,” International Conference on the Applications of Digital Information and Web Technologies(ICADIWT 09), Aug.2009, pp.68-74.

[7] Lazar, I. Parv, B. Motogna, S. “A platform independent component model for dynamic execution environments,” the 10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC 08), Sep.2008, pp 257-264.

[8] Espiritu Jose F, Coit David W. “Component replacement analysis for complex electricity distribution configurations,” IIE Annual Conference and Expo 2008, May.2008, pp.170-175.

[9] Jiaoyan Wang, Fengju Kang, Minghao Liu. “The Technology for Rebuilding 3D Models by Components,” Computer Simulation, vol 26, Oct.2009,pp.252-255(in Chinese).

[10] Fang Yu, Gui-lan Shen, Xie Huang. “Software components assembly based on predictability,” Computer Engineering and Desing, vol 29, Nov. 2008, pp.2970-2972(in Chinese).

[11] Chun-guang PENG, Jian-xing Gong, Ke-di Huang. “Research of Simulation Model Component Assembler Based on BOM,” Journal of System Simulation, vol 20, Dec. 2008, pp. 3175-3178(in Chinese).

[12] Friedhelm Nachreiner, Peter Nickel, Inga Meyer. “Human factors in process control systems: The design of human-machine interfaces, Safety Science,” Volume 44, Issue 1, Safety and Design, January 2006, pp.5-26.

[13] Sotiria Drivalou, Nicolas Marmaras. “Supporting skill-, rule-, and knowledge-based behaviour through an ecological interface: An industry-scale application,” International Journal of Industrial Ergonomics, Volume 39, Issue 6, November 2009, pp.947-965.

[14] Naiyana Tansalark, T.Claypool. “XCompose: An XML-Based Component Composition Framework,” http:// www.cs.iastate. edu/~lumpe/wcl2003/camera/naiyana.tansalarak.pdf.

[15] Ming-chuan Zhang, Hong-yi Wang, Shi-bao Sun. Research on Assembly and Fault-tolerant of Interface Component in Distributed Human-computer Interactive System,” The Third International Symposium Computer Science and Computational Technology(ISCSCT 2010), Aug.2010, pp.417-421.