Research on Component Assembly Environment and Its Implementation Mechanism Based on Software Product Line

Jianli Dong
HuaiHai Institute of Technology, Lianyungang, 222005, P.R.China
dongjl1019@sina.com

Wen Dong
28th Subsection of United Logistics, LanZhou Military Region of PLA, Xi'an, 710043, P.R.China
dongwen19850204@163.com

Xin Chen
Unit 68026 of PLA, Houwuquan-New Village, Lanzhou, 730058, P.R.China
346117382@qq.com

***Abstract*—For solving the key technical challenge and problem in the heterogeneous components assembly and realizing the industrialization production of software products, and to use the producing system and mode of the modern manufacturing industry for reference, a new industrialized PL-ISEE (product line based integrated software engineering environment) model is firstly proposed. And then, through the requirement analysis of the current component assembly technology, a heterogeneous component assembly line suited to the new PL-ISEE and its framework and implementation mechanism are systemically studied and discussed. This component assembly line or assembly environment will become a real industrialized software product line differed from traditional software engineering environments and development methods. Its research and application will help to promote the formation and development of modern software industry.**

***Index Terms*—Software component, Component assembly, Component based software engineering, Software product line, PL-ISEE (product line based integrated software engineering environment)**

I. INTRODUCTION

In recent years, with the growing maturity and application of new technologies such as software architecture, software component, and large-granularity software reuse, the product line based software engineering methodology has aroused the widespread concern and attention in academia and industry, and become a hot and key topic in software engineering field. The software product line is much like the automated assembly line of the modern manufacturing products (such as car and television products). Its advance and study objective are to implement the automated and industrialized mass-customized production of domain specific software products by the component assembling techniques. The product line will be the most ideal way of software production in the past 40 years of software engineering development. It will play a very important role for promoting the development and formation of modern software industry, and produce enormous economic benefits and social benefit [1-4].

At present, the researches of the software engineering based on product line mainly focus on both product line based integrated software engineering environment (PL-ISEE, called assembly line) and component assembly technology. The both studies are confronting with numerous difficulties. Because existing practical industrialized PL-ISEE products are almost blank in the current software engineering field, and a number of problems remain unsolved in software heterogeneous components assembling process. In the paper, to use the modern industrial product line for reference, a new industrialized component assembly environment (assembly line) and its implementation mechanism will be introduced and discussed. The aim is to explore a new approach for realizing the industrialization production of software products.

II. RESEARCH STATUS OF COMPONENTS ASSEMBLY TECHNOLOGY

Component-based software engineering methodology is a brand new software engineering paradigm (also called the industrialized software engineering paradigm) to take software component, architecture and reuse technology as the base, the application of domain engineering and software product line as the guidance, and the industrialization production of software products as the goal. As research on the new engineering paradigm, the most basic technology is a componentized technology which includes component model and component assembly as well as assembly line/environment. The component model is often used to define and describe the internal organizational structure

doi:10.4304/jsw.8.11.2775-2783

and external behavior characteristic of a component, it is also a template and norm to produce and use the component. Of course, the ideal component model should be a unified standard model defined by the software engineering standardization organization. The standard components developed by using the standard model will certainly bring a great change to truly realize the assembly line and industrialized production of software products.

Unfortunately, the research and development of the component and its model have not achieved unity and standardization whether in academia or in the business community, which also has brought many problems to the production and application of software components. For example, the usual component model classifications include: Component Description Model--REBOOT, ALOAF; Component Protocol Model--3C model, CDL/ADL model (such as ACME, C2, Darwin, UniCon, Wright); Component Implementation Model--CORBA, COM/DCOM, EJB, etc.. As currently widely used component implementation models, the heterogeneous problems between components should be first solved as assembling an application software system with the different model components.

Actually, component assembly coordinates their actions with the relationship between them, and organizes them into an organic whole which is called an application software system. In current component assembly technology research, which can be divided into three kinds of assembly methods--the black-box, white-box and gray-box assembly according to the assembling characteristics formed by the understanding degree of the component internal details in the assembly process[5-6].

The black-box assembly method is the best method. It neither require any understanding of the internal implementation details of the assembling components and nor do any configuration and modification to them. Black-Box assembly method takes the component as a black box do not known the internal structure, and directly assemble the components in accordance with their functions and interface constraints and realize application software's production. But it indicates that the black-box assembly method requires "the invariable components to be assembled into the ever-changing application system", and uses a limited component sets and assembly operation to generate unlimited and evolving application sets. In fact, this assembly method must require that the limited component sets should have a high spreadability, completeness and inclusiveness on the components demanded by all application software. This is asking too much of components to be achieved with existing component technology. It is just the problems facing blank-box assembly.

In contrast to the black-box assembly method, the white-box assembly method regard component as a fully transparent white box, all the implementation details of the component are displayed. Developers can implement components assembly under fully understanding the premise of the internal control logics and the calculation flows of the component. And besides, they can also modify and adjust component features according to the application requirements and then assemble them. Obviously, the software components to be freely modified are not in the true sense of the reusable products, and will lead to component's security, reliability, standardization, quality characteristics to be never guaranteed. To make matters worse, the understanding and modification of a component internal implementation details are often much higher than the development cost of a new component. These are the problems facing white-box assembly [7].

The gray-box assembly, to be intervenient between black-box and white-box, is to meet the needs of the software assembly by adjusting component assembly mechanism and rather than modifying components themselves. This method not only achieves the flexibility of component assembly, but also let the assembly process not to be too complex. Therefore, the gray-box assembly method has become the mainstream direction and reasonable choice of component assembly technology in recent years. But the main problem facing the gray-box assembly method is the heterogeneity of different model components. So, the gray-box assembly method researches are mainly centered in component wrapping technology and assembly environment (also called component assembly line) development. The research objectives of the component wrapping technology are to wrap all heterogeneous components with a unified component model (called wrapper) and to shield the heterogeneity between different components. And study goals of the assembly environment are to provide a software assembly line resembling an industrial product line for assembling various model components and realize the industrialized productions of application software products finally.

This article will systemically research and discuss the integrated assembly environment and its implementation mechanism based on the gray-box assembly method. The environment and facilities are also called industrialized software product line or software assembly line.

III. Component Assembly Environment and Its Implementation Mechanism

To realize heterogeneous components assembly, the problems of building component assembly environment (called as assembly line) and removing components' heterogeneity are firstly researched and solved. The problem about the removing heterogeneity has been discussed in other relational articles. Here, we mainly research and solve the problems of component assembly environment and its implementation based on software product line.

A. A New Industrialized PL-ISEE Architecture Model

The component-based industrialized software engineering paradigm emphasizes producing software products by the method assembling components rather than the traditional programming method from scratch. And it also emphasizes that its software production

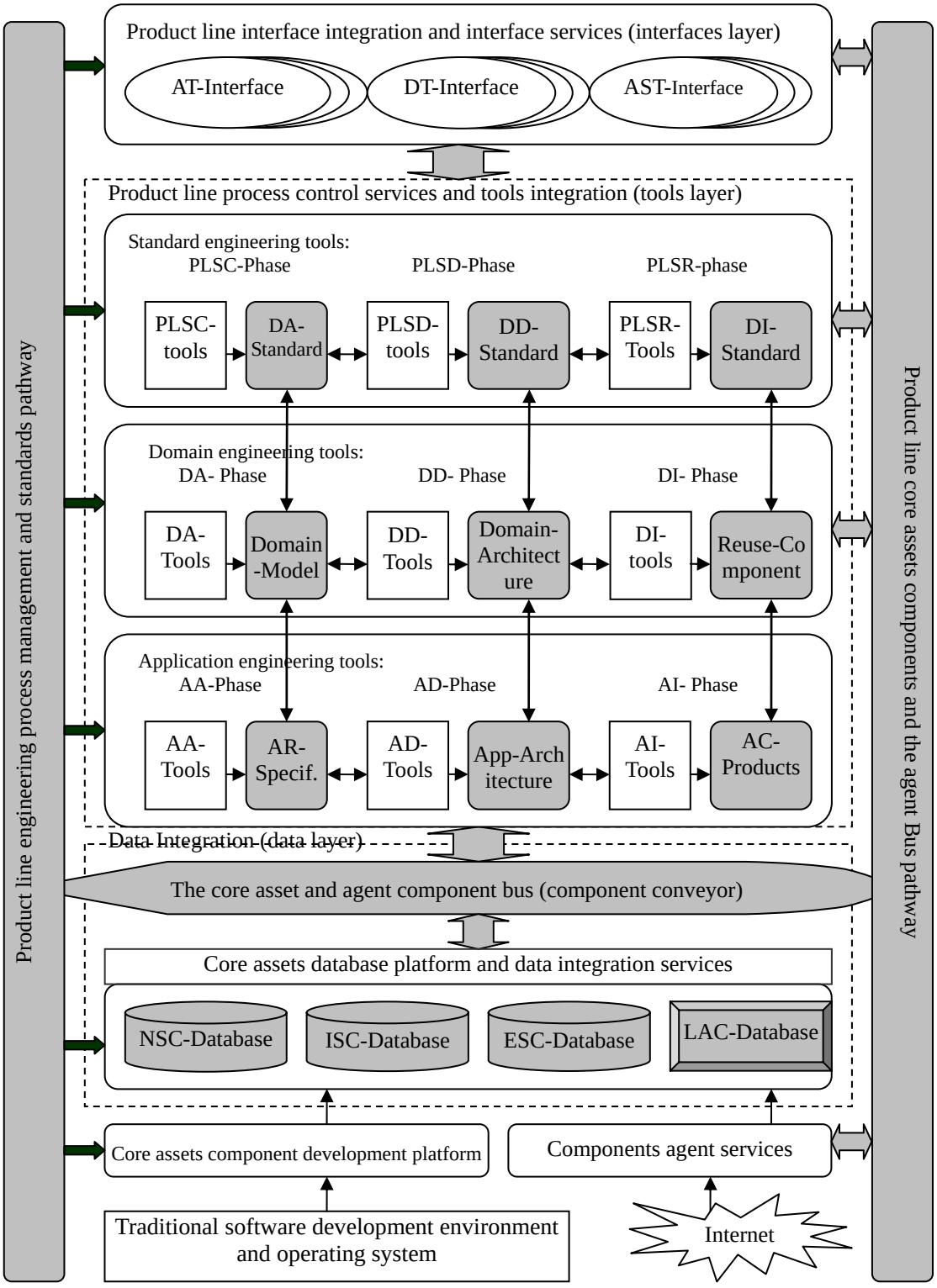

Figure 1. A new industrialized PL-ISEE architecture model

process and life cycle model is completely different from the traditional software engineering one. Its life cycle is defined as: 1) Requirements analysis; 2) The design and evaluation of software architecture; 3) Components selection and customizing; 4) Components assembly and system configuration; 5) System testing; 6) Software application and maintenance. The components assembly takes the system architecture and framework as design blueprint, and assembles and configures the selected and customized components to generate the application system. It shows that component-based software development method is another new software engineering methodology aimed at realizing the industrialization production of software products after

the traditional structured software engineering methodology and modern object-oriented software engineering one. However, we still face many problems to translate the new methodology into reality and produce software products in the COTS (Commercial Off-The-Shelf) component assembly method. One of them is how to build an ideal industrialized component assembly environment (assembly line). This environment is also called as the product line based integrated software engineering environment in the software (PL-ISEE).

The research and creation of PL-ISEE model must be on the basis of correct and complete product line based software engineering process models and life-cycle model. This process and life-cycle model is adopted to define and describe all the production process and requirements, such as activity sequences, task frameworks, the technology methods, standard specifications, quality controls, asset components structures and so on in product line software development. It is consider as the behavior guideline and activities norms of realizing software product line engineering and its products production, and has been the pre-conditions and important foundation of PL-ISEE research. Withal, we have proposed an opened "N-Life Cycle Model" with modern manufacturing industry features in article [8]. According to the opened "N-Life Cycle Model", a new industrialized PL-ISEE architecture model is created and shown in Figure 1.

This new industrialized PL-ISEE model as shown in Figure 1 is proposed based on the unified product line engineering concept model, large granularity reusable asset data model, component assembly behavior model, and core assets development and application software production iterated evolution model. The framework of this new environment model is essentially a three-layer (interface, tool and data layers) architecture model with core asset and agent components as the software bus. On the bus or data layer, it is a PL-ISEE similar to an industrialized product line of the modern manufacturing industry, which will realize the assembling production of software product. All the components needed on software assembly line can be provided by core asset and agent components bus as a component conveyor belt in the data layer; under the bus or data layer, it is the traditional ISEE based on traditional software engineering methods, which support the developments of source program and documents of the core asset components. Obviously, the new framework model is a real PL-ISEE model with the software industrialized producing (assembling) ability and control mechanism which is very similar to the automated assembly line and management mechanism of the modern manufacturing industry [9].

Figure 1 Notes: Filled boxes represent the products of the process or tools or phases. No filled boxes represent the process or tools.

Product Line Interface Integration and Interface Services (interface layer): AT-Analysis Tools Interface, DT-Design Tools Interface, AST-ASsembly Tools Interface.

Standard Engineering Tools: PLSC-Product Line Standard Classification, PLSD-Product Line Standard Design, PLSR-Product Line Standard Release; DA-Domain Analysis Standard, DD-Domain Design Standard, DI-Domain Implementing Standard.

Domain Engineering Tools: DA-Domain Analysis, DD-Domain Design, DI-Domain Implementation.

Application Engineering Tools: AA-Application Analysis, AD-Application Design, AI-Application Implementation.

Core Assets Database Platform and Data Integration Services: NSC-National Standard Components, ISC-Industry Standard Components, ESC-Enterprise Standard Components, LAC-Local Agent Components

B. Heterogeneity Problems of Components Assembly in the PL-ISEE

To realize the industrialized assembling production of application software products by the component-based software engineering (CBSE) and software product line methods, it has two basic conditions: Unified component model and integrated component assembly environment (assembly line). The component model defines the internal structure and external features of a component. External features provide the component interfaces and how interactive mode with other components; assembly environment is an integrated software development environment and infrastructure which support component assembly process and application system assembly production, such as the PL-ISEE shown as in figure 1 is just a new component assembly environment proposed by the authors [9-10].

The basis of implementing component assembly must conform to a unified component model and standard. Unfortunately, component's development and application are still far from reaching this requirement. Currently, the component implementation models which have been universally acknowledged by software engineering field include three types: OMG organization's CORBA model, Microsoft's COM/DCOM model and SUN Company's JavaBeans/EJB model. But each company goes his own way, and leads to the different component models and standards. This brought huge challenges to the components assembly [11].

Different component models and their corresponding component products will lead to the heterogeneity and non-interoperability issues between the components. Heterogeneity mainly represents two aspects: the first is component model's heterogeneity, which limits that application developers can only use the components from the same component model. While others model components having the same functions are used, these components must be redeveloped in same model. But this redeveloping work certainly increases the cost of software development. The second is the heterogeneity of component running platform or environment, which makes that the application system assembled by using one model's components can not run under the platform

and environment supported another heterogeneous component model. It can also make the component assembly suffered a serious setback. Precisely, due to the emergence of heterogeneous components, the component assembly will be confronted with many difficulties as follows [12]:

1) For a similar concept and behavior, the different component model provides different accessing modes. For example, in the JavaBean system, access to the component attribute is achieved by declaring a public method member's name. In the COM, however, it is not dependent on its method member's name, but is realized by method sequence in component's binary interfaces. Additionally, different component models provide different component instantiation processes.
2) The same component model can be implemented by different programming languages. This will likely lead to inaccessible components belonging to the same one component model. What is more serious is that the data types and data structures defined and used in a component in a language are not directly supported by other programming languages.
3) Different component model provide the different persistence mechanism and method of the component. This will imply that an application system assembled with the heterogeneous components may produce inconsistent data storage structures or database schemata during operation, Such as RDB and OODB schema. It is absolutely not allowed in the same application system.
4) The graphical identities of different components depend on the different operation platforms or running environments. Such as the JavaBean components' visualization needs a virtual Java machine and graphics container, but COM ones need the window handles of the Windows Operation System.
5) Heterogeneous component assembly requires developers to be very familiar with all component models. In fact, is very difficult that the different types of application systems are developed by the different component technologies.
6) The manufacturers (or enterprises) with different component models use or provide respective component development environments and tools. But the environments and tools may be incompatible or unavailable each other. This phenomenon has also increased the difficulty of assembling application system with heterogeneous components.

Above the heterogeneity problems of components assembly seriously hinder the development of software product line. To implement heterogeneous components' assembly, component's heterogeneity problems must be removed and solved. At present, component's heterogeneity can be removed by wrapping heterogeneous components with a unified component wrapper. That is to say, wrapped components will become no-heterogeneous standardized components which are can be directly used and assembled into an application system. About the problems of the component will be further discussed in other papers.

C. The Features and Demands of Component Assembly Environment

Components assembly is the most basic operation to achieve the software production based on software component and product line. The assembly process should mainly include the operations of the components' selection, browsing, graphical representation and visualization, wrapping, assembly and configuration, and deployment and operation. In order to the effective implementation of these operations and application system assembly work, we need an integrated component assembly environment which can meet all the assembling process and operation needs. This environment should include the functions and tools as follows [15-16]:

1) ***Component browser:*** In the assembly process, the developers should be able to browse the selected component information which includes component domain, level, attributes, distribution, quality, and so on. The component and its information are usually stored in the components database or product line assets database. Developers can find all components met the conditions according to the system design requirements. A design requirement may be a function demand focused on the component operations, it may also be an implementation structure demand mainly focused on some attributes of the component, or it may be a domain demand focused on the engineering domain characteristics of components. For the components met the conditions, they will be instantiated according to the distributed information of the components and then displayed in the assembling line by the browser.
2) ***Component Graphical representation:*** In order to facilitate operating and understanding the components in software design process, they should be represented in a graphical manner in the component assembly scene. The graphical representation of the component should be equivalent to the text expression (such as the components described by using component description language) of ones. The graphical representations can highlight the information being most concerned by developer, and the unimportant information is shielded, which help to enhance the visualization ability of component representation and assembly, and reduce the complex degree of the software development process.
3) ***Components coherence expression:*** In order to shield the component differences from different component models for the application developers, the assembly environment should provide the corresponding wrappers for all the different model's components. And all the wrapped heterogeneous components can provide the same external interfaces and present the consistent views of the components in accordance with the united wrapping model. Furthermore, a generic component

model can be abstracted from the combination of multiple wrappers, and it can provide a mapping mechanism from the wrapped component to the actual component. From which we can see that the wrapping model is actually a standardized template to achieve the uniform and coherence representation of heterogeneous components, and it is also the key to realize heterogeneous components assembly.

4) ***Component connector:*** The interactions between components play a very important role in the component assembly process. The purpose of component assembly is to achieve interactive and data transmission between the different components according to software architecture needs. The connector is a tie or a bridge to achieve the interoperations and communications between the different components. In the components assembly process, the developer should determine the interactive relationships between the selected components according to the system architecture or framework, and then choose the right connectors satisfied the relationships to realize the connections between the selected components. The connector types have pipe type, method invocation type, event connection type, C/S communication protocol type, and so on.
5) ***Component assembly description language:*** The component assembly environment should show the processes of the selecting, connecting and assembling components for the application developers who know the architecture and assembling progress of the whole system at any time. Therefore, the components assembly description language (CADL) which describes the process and status of components assembly is required. It can clearly describe the current assembling state of the application system. The assembling state is actually the connection relationship between the system components, connectors and their mutual. And then, the CADL can be also used to record and describe the deployments and evolution demands of the application system.
6) ***Component database management system:*** In the heterogeneous components assembly process, the selection, wrapping, connection, assembly and deployment of components are the basic operations to develop the application systems. The implementation of the operations is centered on the components database. Therefore, a successful components assembly environment must provide a component database management system which can meet all the component operations. The biggest differences between component database management system and general DBMS is that the schema and view designs of the component database must be centered on the component operations and managements [17-18]. In fact, a well-designed component database management system should have all the operating functions and tools configuration about above-mentioned search, storage, selection, browsing, graphical and coherence representation of the components.

D. Framework and Configuration of Components Assembly Environment

The architecture of heterogeneous components assembly environment is shown in Figure 2 [19-20]. The component assembly based integrated software development environment uses a hierarchical architecture. And it consists of the interface layer, tool layer, assembly layer and component layer (also called data layer) from top to bottom. The tool layer integrates all operating tools needed for assembly and software development process, the interface layer integrates the user interfaces of a variety of supporting tools (also called the environment tools), and users can implement various operations through the interfaces calling tools. Of course, the interface layer should also provide the scheduling, monitoring and management facilities of the assembly tasks. In the assembly environment, the tool layer integrates all operation tools above section described, such as the search, selection, browsing, graphical representation, assembly, configuration, deployment, and operation of the components, its purpose is to provide developers with a convenient and efficient visualization software assembly production environment. The assembly layer (known as the assembly line) is a site to implement components assembly and application system production. The data layer is components' providers and managers which can use and manage all components through the component database management system.

The core constituent parts of the assembly environment as shown in figure 2 should include: component wrapper, wrapper base class model, connector, component assembling workshop (or assembling line or assembly scenario), supporting tools set and user interfaces set, and component database system. The main functions of the environment constituent elements are outlined below:

1) Component wrappers (in the assembly layer in Figure 2) encapsulate (or wrap) the actual components corresponding to various component models by the object oriented inheritance mechanism and method as well as association relation, and provide a unified external interface. That is that all heterogeneous components wrapped by the wrapper will provide a consistent component view. In fact, the component wrapper is a standardized template to be used to package and represent a variety of heterogeneous components in accordance with a unified component model. The common features abstracted from all the different heterogeneous component wrappers will generate a wrapper base class model. And then, the unified public interface and feature set of the wrapper are defined in the base class. The component wrapping model both has realized a unified access method to

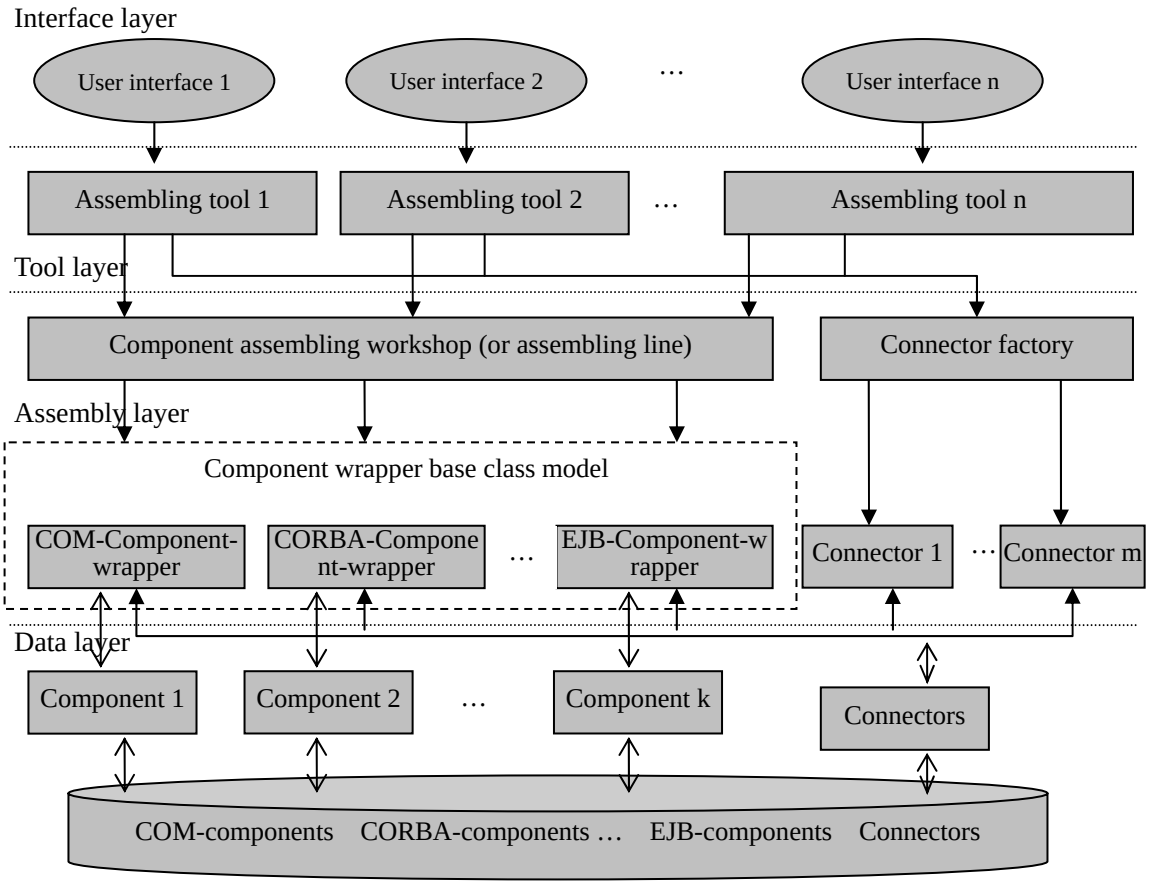

Figure 2. Framework of heterogeneous component assembly environment

specific heterogeneous components and also provided a mapping mechanism to a specific component model.

2) Connector (in the assembly layer in Figure 2) is one of the important elements for implementing the components assembly (the connectors and components are generally called components). The component connector is the bond and bridge connecting components. The purpose is to achieve the interactive operations and communications between the components. Developers should determine the suitable connector types according to the correlative relations between the components in software architecture, and then the connectors met the types are produced and provided by the connecter factory. The connectors are instantiated into the connecting components matching with the needed connecting relations. In Figure 2, the developers select the required components in the component database and put them into the component assembling line. And then developers can read and see the component entities and their related information in detail by using browser or graphical representation tools provided by the environment. If found a component to interact with another one (such as a method call, the service request and communication, etc.), then the corresponding connector and connection type between the two components will be established according to their interactive relations.

3) Component assembly line/workshop (in the assembly layer in Figure 2) which is generally called software product line or assembly line is a workshop to assemble components and generate final application system. When developers want to assemble components, they can access an assembly workshop or an assembly line by calling the assembly workshop interface or assembly line interface from interface layer to tool layer, and then implement all the operations of the selection, assembly, configuration, deploying, and instantiation of the components in the assembly environment by a visual way. For the components, an assembling workshop is just a component container. Actually, the assembling environment can provide all operations of the component assembly through interface layer calling tool layer. Usually, the assembly workshop should provide some graphical tools which can implement the graphical representations of components and their interrelationships as well as all operations in the workshop. In order to facilitate the rapid deployment of a component-based application system and understand the configuration information of the assembled components, the developers can create an assembling process description document by the component assembly description language integrated in the workshop. This document can record and describe the assembly process and configuration information of the application system in text mode.

4) The components database system (in the data layer in Figure 2) provides the storage and management of the components (including connectors, frameworks, architectures, etc.). In integrated components assembly environment, the component database management system should be an important infrastructural construction to realize the industrialized production of software products. Compared with the design of the conventional database, the designs of the data model, database schema and application view of the components database system are very difficulty and special. Component library should be designed based on a component model, the wrapper model, the software architecture and framework model as well as component assembly model, so as to meet the functional and performance requirements of the modeling, representation, assembling, storage and management of the various heterogeneous components. About the design and implementation theory of the component database system, we have discussed and researched in the related papers.

Here, we can clearly see that above component assembly line is a complex software engineering supporting environment similar to modern manufacturing industry product line (such as TV and Car production lines). Its objectives are to realize the industrialized production of software products by using software components as constituent elements. So, the assembly environment is completely different from traditional software engineering environment. The former provides a new industrialized production mode, and the latter only does traditional manual development mode.

IV. Conclusions

Component assembly is the key technology of the component based software engineering methodology researches, which has draw much attentions of academia and industrial circles. The component assembly study consists of two parts: The study of the formation process of the composite components, and the study of architecture-oriented (or framework oriented) components assembly strategy. The purpose of the former is to get large granularity composite components which own stable performance, strong reliability and high reusability, so as to facilitate components' assembly and application. The latter is to seek out a good assembly strategy to build application systems. This paper has conducted the study of heterogeneous component assembly technology from the aspects of component wrapping, component assembly and assembly environment framework. We hope to have explored a new feasible way for the current component-based software engineering methodology and assembly-line based industrialized and automated production of software product. The heterogeneous component assembly line theory, technology and environment proposed in the paper are studying for this goal [21-23].

The biggest difference between component-based software engineering methodology and traditional structured or object-oriented software engineering ones is the industrialized production of the software products. Thus, component-based software engineering methodology should include the studies of component assembly technology, software architecture, and product line based integrated software engineering environment. Although the paper has proposed new technology and idea for assembling heterogeneous components, but we have a long way to go for exploring how to form the mature industrialized software engineering environment and product line.

Acknowledgment

This project is supported by Fund of Jiangsu University Natural Science Basic Research Project, Grant No. 08KJD520013 and Jiangsu Key Built Discipline Project—Computer Application Technology.

References

[1] YANG FuQing, "Thinking on the Development of Software Engineering Technology", *JOURNAL OF SOFTWARE / RanJian XueBao,* Vol.16, No.1, pp.1-7, 2005.

[2] ZHANG Yousheng, LI Xiong, *Software Architecture Principle, Method and Practice*, Beijing: Tsinghua Press, 2009.

[3] Zhang Xinyu, Zheng Li, Sun Cheng, "The research of the component-based software engineering", *In Proceedings of the 6th International Conference on Information Technology*, pp.1590-1591, 2009.

[4] YANG FuQing, LV Jian, MEI Hong, "Internetware technology System: A Approach centered Architecture", *Science in China (Series E: Information Sciences)*, Vol.38, No.6, pp.818-828, 2006.

[5] SHEN Li-wei, PENG Xin, ZHAO Wen-yun, "Software Product Line Architecture Modeling and Component Composition Implementation with Extension of Aspectual Mechanism", *Chinese Journal of Electronics,* Vol.37, No.4A, pp140-145, 2009.

[6] XIE Wu-ping, XUE Jin-yun, WAN Song-song, Aspect-Based Component Model and Its Assembly and Implementation, *Computer Technology and Development*, Vol.19, No.4, pp.160-160, 2009.

[7] Zhijian WANG, Yukui FEI, Yuanqing LOU, *Software Component technology and Application.* Beijing : Science China Press, 2005.

[8] Jianli DONG, "Research on software engineering process model based on software product line architecture", *Computer Engineering and Design,* Vol.29, No.12, pp.3016-3018, 2008.

[9] Jainli DONG, Ningguo SHI, "Research on the CORBA Implementation Mechanism of a New Industrialized PL-ISEE", *Journal of Software, Academy Publisher,* Vol.7, No.5, pp.1171-1176, 2012.

[10] YE Junmin, CHEN Zhuo, LEI Zhixiang, YE Yanfeng, ZHAN Zemei, "Research on application development process based on component composition", *Application Research of Computers*, Vol.25, No.6, pp.1736-1738, 2008.

[11] Basem Y Alkazemi, "A Precise Characterization of Software Component Interfaces", *Journal of Software, Academy Publisher,* Vol.6, No.3, pp.349-365, 2011.

[12] Longye Tang, Yukui Fei, Zhijian Wang, "Service-Oriented Component Model", *International*

Journal of Advancements in Computing Technology, AICIT, Vol.3, No.1, pp.68-79, 2011.
[13] Sajjad Mahmood, Azhar Khan, “An industrial study on the importance of software component documentation: A system integrator’s perspective”, *Information Processing Letters*, Vol.111, Issue 12, pp.583-59, 2011.
[14] Basem Y Alkazemi, “A Precise Characterization of Software Component Interfaces”, *Journal of Software, Academy Publisher*, Vol.6, No.3, pp.349-365, 2011.
[15] Jianli DONG, Ningguo SHI. “A study on framework and realizing mechanism of ISEE based on product line”, *Journal of Software, Academy Publisher*, Vol.5, No.10, p.1077-1083, 2010.
[16] HE Tian-zhang, WANG Pan-qing, LI Xiao-hui, “Research and Application on CORBA Component Assembly”, *Science Technology and Engineering*, Vol.8, No.1, pp.84-86, 2008.
[17] Jianli DONG, “Framework and Schema Design of A New Industrialized PL-ISEE Database Supporting Platform”, *Advances in Information Sciences and Service Sciences*, AICIT, Vol.4, No.17, pp.324-332, 2012.
[18] Jianli DONG, Ningguo SHI, Yanyan CHEN, “Research on Framework and Realizing Mechanism of a New Industrialized PL-ISEE Database Supporting Platform”, *JCIT, AICIT*, Vol.7, No.13, pp.197-205, 2012.
[19] MAO Yingchi, LIANG Yi, WANG Zhijian, “Design and Implementation of Model for Heterogeneous Software Component Composition”, *Computer Engineering*, Vol.31, No.4, pp.56-57, 2005.
[20] CHANG Bing-guo,WANG Xiang-zong, “Research and development of component integration support platform”, *Computer Engineering and Design*, Vol.32, No.8, pp.2712-2715, 2011.
[21] Sajjad Mahmood, Azhar Khan, “An industrial study on the importance of software component documentation: A system integrator’s perspective”, *Information Processing Letters*, Vol.111, Issue 12, pp.583-59, 2011.
[22] Zhaohui Li, Lixin Shen, Haijun Mao, Ying Zhang, Ting Xu, "A Component Based Agile Software Configuration Development Method", *International Journal of Advancements in Computing Technology, AICIT*, Vol. 4, No. 11, pp. 405-413, 2012.
[23] Xianjun Li, Gang Ye, Zhongwen Li, Shilong Ma, “Study and Implementation of Spacecraft Integration Test Platform Based on Component Technology”, *Journal of Computers, Academy Publisher*, Vol.6, No.5, pp.963-968, 2011.

Jianli DONG was born in Shanxi province, China, in 1957. He got his Bachelor of Mathematics Science in Northwest Normal University, Lanzhou, Gansu province, China, in 1988 and got his Master of Software Engineering in Beijing University of Aeronautics and Astronautics, Beijing, China, in 1995. He is now a professor at the School of Computer Engineering in HuaiHai Institute Technology, Lianyungang, China. He has published over 80 papers, and completed over 8 scientific research projects, and won 6 times scientific and technological progress awards from the province and military.

Mr. DONG current research interests include software engineering, integrated software engineering environment, software architecture, engineering database system, and object-oriented technology.

Wen DONG was born in Shanxi province, China, in 1985. He got his Bachelor of Communication Engineering in Xi’an Communication University of PLA, Shanxi province, China, in 2005. She is now a communication engineer at the 28th Subsection of United Logistics, LanZhou Military Region of PLA.

Ms. DONG current research interests include communication engineering, computer application technology, engineering database system, and software application.