A Component-based Software Development Method Combined with Enterprise Architecture*

Luo Yi[1], Wu Chanle[1,2,3], Huang Lei[1] and Ye Gang[3]
[1]School of Computer, Wuhan University, Wuhan, China
[2]The State Key Laboratory of Software Engineering, Wuhan, China
[3]National Engineering Research Center of Multimedia Software, Wuhan, China
luoyi9999@hotmail.com

Abstract – It's very important to ensure the unity of software architecture and business goal in application software development. This paper presents a new software development method based on enterprise architecture, describes the process of building enterprise IT architecture and transformation enterprise IT architecture into specific software architecture. It also establishes a framework of lifecycle application development platform around with architecture, and practices this method in the software development department of a certain large enterprise. The practical results show that the overall defect rate of releases decreases significantly.

Index Terms - Enterprise Architecture, Component-based Software Development, Software Product Line, Architecture Governance

1. Introduction

With the development and maturation of component-based technologies, the component-based software development methodology (CBSD) has been widely promoted, and has been widely used in the industrial sector [1][2] in recent years. The CBSD method makes complex software development into component development and flexible call or combination of components through the middleware technology, which greatly improves the efficiency of software development and reduces its complexity. In the CBSD development method, software architecture (SA) is the guiding framework and portfolio constraints for development and combination of components, therefore researchers put increasing emphasis on the role of software architecture in the CBSD development [2]. However, in the process of large-scale enterprise-level application software development, emphasizing only the software architecture of a single system is not enough. It is very important to ensure the unity of enterprise-level application software architecture and business goals, and to implement enterprise-level component-based software development.

2. Enterprise Architecture

Zachman proposed the concept of Enterprise Architecture (EA) for the first time in 1987[3]. Enterprise Architecture is a methodology of planning IT systems from the overall perspective of the enterprise, and is also the bridges and means that convert enterprise business architecture (BA) into the IT infrastructure. In recent years, with the deepening of globalization, people takes more attention on building customer-centric IT systems and overall corporate strategy planning enterprise's IT systems, with no longer on a single product perspective of IT systems, So EA method has also been widely used and improved. ZIFA, GARTER, IBM, TOGAF[4] established the complete methodology of EA[5]. EA divides the construction of enterprise IT systems into four views: business architecture, application architecture, data architecture and technical architecture, as shown in Fig.1.

The business architecture is a collection of business organizations, processes and functions which are carried out around the enterprise strategic layout. It covers the information models, function models, process models of the enterprise business processing and the corresponding organizational structure and role definition. The application architecture is the organization, configuration and relationship constraints of application functionalities of multiple IT systems within the enterprise from the functional perspective. The data architecture is classification of information, data of the enterprise IT system, and unified organization and description of information processing and information exchange interface from the information perspective. The technical architecture is the all IT infrastructure needed to help construction of related systems from the IT implementation perspective.

When EA is used, we need stepwise decompose the business process in accordance with the functional module until a certain feature of the process can be implemented through an application system function, and then refine and re-divided those function module in accordance with certain application principles. The result is the functional partitioning of the application system and process relationships. Similarly, we classify and subtotal all information needing business processes to refine conceptual data model of the enterprise information processing, and then detail the conceptual data model following the application of the subject and classification, to form the logical data model and the physical data model of a certain application system. After the application architecture and data architecture determined, the last is to choose operating system, database, and plan and configure those network middleware that application systems needed and form the technical architecture of the EA. The

* This work is supported by the major national science and technology special projects (2010ZX03004-003-03) and the fundamental research funds for the central universities (3105005, 111087).

above is the EA process that transform business architecture to IT architecture.

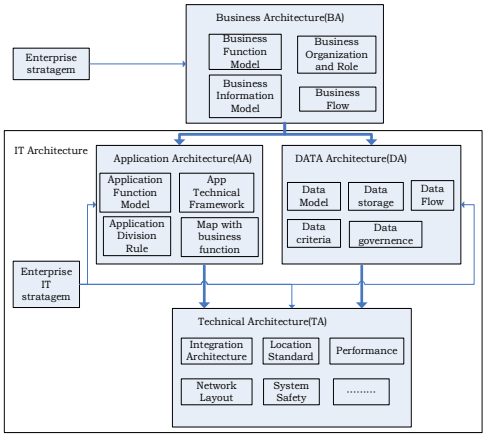

Fig.1 Overview of Enterprise Architecture

3. A Component-based Development Methodology Combined with Enterprise Architecture

With the maturity of object-oriented technology and the Web Service technology, the procedural structured coding process gradually evolved into the assembly process of those service components. The component-based software development method is generally adopted by the industry. How to combine with the existing component-based software development method in a large-scale enterprise when we import the EA method? The following is the practice that the author explored in the software development sector of a large enterprise.

A. Establishment of the enterprise IT architecture through the EA process

In the EA process, the clean-up process of business architecture and the abstract extraction process of application architecture, data architecture, should be consistent with certain principles of components. The similar application functions should be integrated. At the same time, those basic shareable features should be extracted in accordance with the principle of separation of concerns. We should guarantee the function module within a hierarchy to be simple and be shared in accordance with the hierarchical principle.

(1) Application Architecture

Fig.2 shows the application architecture of a large-scale enterprise in accordance with the above principles. In addition to the consideration of component-based software development, the formation of the application architecture take into account with certain business principles such as customer-centric, flexible product assembly. Application Architecture divided the entire enterprise application system into four layers: the channels layer is the access point that accepting customers' request. A variety of customer service channels belong to this area; the interactive control layer, as the bridge between the channel access and business processing; the business process layer contains the enterprise's products and services. In accordance with the principle of shared services, the business processing layer is further divided different modules such as Customer Information, Products and Services, Business Support; the decision analysis layer accepts information and data from the front layers, carries out analysis and processing, and forms the business statements and specific topic-oriented data marts.

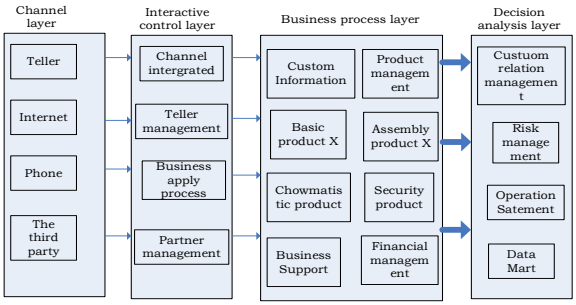

Fig.2 Example of application architecture

(2) Data Architecture

Fig.3 shows an example of the conceptual data model in data architecture. As the result of further abstracting of the information entity required to establish the customer services, the seven information subject fields cover all the information needed in the day-to-day running of the enterprise. In this example, a variety of services that enterprise provides to customers can be described using the product theme. It signs a product agreement with a certain customer group through some channels to provide a specific service instance.

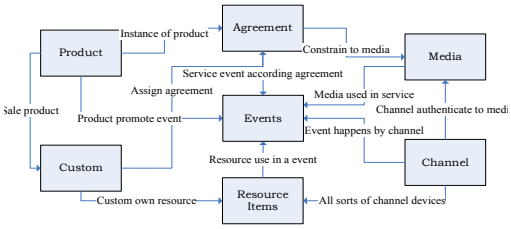

Fig.3 Example of conceptual data model in data architecture

(3) Technical Architecture

Determination of the technical architecture should consider not only the functional requirements of the application architecture and data architecture, but also the overall enterprise IT strategies and the technology roadmap, such as the decentralized or centralized system. Based on this point, we should consider from three aspects. First, making choice of Service Bus, scilicet, exploring how all types of the heterogeneous application systems implement the service-oriented architecture, and achieve the channels end call for all kinds of services and ensure consistency; Second, from the perspective of process integration, determining of the coordination mechanism that between the business partners systems and the internal applications, including automated processing and workflow platform; Third, from perspective of data integration, determining the basic architecture of data transmission, conversion, processing between applications systems. Finally, constructing the overall IT infrastructure taking into

account network bandwidth, equipment selection, system throughput, and other factors structure or exploring the part needed to be enhanced, such as service bus.

B. *From the enterprise IT architecture to the specific system architecture*

Facing the establishment of application system in a specific area, it is necessary to implement the architecture design referring to the enterprise IT architecture.

(1) Position the layer of the application architecture on which the system lays and classify functional modules in terms of the domain system's demand analysis. For example, when the enterprise needs to provide customers with certain types of new product, the system can be positioned in the business processing layer of Fig.2 according to the demand to add a certain type of assembly products; then in the light of relationship among the application systems of Fig.2, further analysis is made on relations between the required basic products and their assembly; at the same time according to the requirement of the product in the channel service (for example, sale the product by telephone), improve function for a specific channel system. In terms of the reports requirement of the product, supplement a certain reporting system of decision support layer or a certain data mart.

(2) Determine the required data and information elements in terms of the enterprise data architecture, establish a correlation with the enterprise conceptual data model, and determine the logical model of relevant data, data transfer standards for interacting with other systems and data lifecycle.

(3) Determine the technical standards according to the enterprise technical architecture, including choices of the hardware, network platform, operating system, database and related middleware which should be used to construct the system, and how to use the existing IT infrastructure.

(4) The design of the product includes both the new features of the product system on the business process layer and improvement on the channel layer and decision analysis layer. All must be designed to meet explicit functional needs of the product and potential of non-functional requirements.

C. *Software product line based on Enterprise Architecture*

In the CBSD development methodology, software product line includes development personnel, technology, platforms, components and all sorts of assets related to a specific area. The greatest effect of the product line is integrating and sharing of resources within a specific area [6], and keeping long-term optimization and maintainability of the corresponding product systems through product line governance perspective.

As for large-scale enterprises, application software development teams need a collection of various branches of the internal sectors. Due to the geographical and cultural distribution of enterprises, the development teams are cross-region and cross-sector. In the development method based on enterprise architecture, it is easy to find the way to divide overall corporate IT system into a number of product lines through hierarchical division and classification of IT application architecture. Due to the explicit definition of the relationship among the application systems, the division of these product lines can even scattered in various regions of the enterprise. They work together in accordance with the holistic view of the enterprise architecture.

For instance shown in Fig.2, there are two basic division approaches of product lines can be considered. The first approach: dividing the product line in accordance with the application architecture layers, such as channels and interactive control, business processing and decision analysis layer. The benefits of this division is that certain types of technically similar systems are divided into the same product line, which technically conducive to promote technology sharing and upgrade product line. Such as the establishment, refining and sharing of channels of product lines can quickly promote the channels related technology and components, and enterprise service capabilities based on these assets can be expanded from a channel to another channel.

The second approach: dividing the product line in accordance with the characteristics of business products themself in business processing layer, including the functional logic of the product, the front channels, background analysis, such as the line of financial management, security product. The benefit of this approach is that the divided product lines agree with external service, are conducive to capability improving rapidly of a single internal enterprise service with the unified IT system processing. But this approach is not fully shared at the technical level, and may even cause the waste on the local system construction. The actual dividing program of product line is on the basis of combination of these two approaches flexibly.

D. *Governance and maintenance of EA*

As for development methods based on enterprise architecture, many enterprises did not carry out overall IT planning in accordance with the thinking of EA at the beginning, They were forced to use the method when enterprise IT systems grow to a certain stage, or even to the disastrous situation. The establishment of the application architecture, data architecture, technical architecture, is just the beginning and not the end. How to reshape related product lines by the holistic view established in accordance with the enterprise architecture, re-positioning system, carrying out the corresponding transformation, promoting the overall enterprise architecture governance and structure transformation, are the difficulties of EA implementation.

In general, while promoting architecture governance, projects from the development of the business itself comes one by one. How to deal with the relationship between the two sides is the key to promote successful transformation of the architecture. It can be taken in two modes. One is to make a statement of the overall enterprise architecture including business architecture to the enterprise decision-making layer, and explain the pros and cons of interest caused by the governance and transformation of architecture. With the help of the decision-making power, obtain a one-off legal status and time window of architecture governance and transformation,

then follow the overall analysis on transformation of the enterprise architecture to determine key points which each application system and the overall IT infrastructure need to reform. Each product line is launched from the top down, to ensure that the one-off and holistic architecture transformation is successful.

The other mode is more realistic. Decompose the architecture transformation points one by one in terms of application systems, and then combine with various project requirements which come from business innovation and development to form various project groups. Then architecture transformation combines with business development gradually. There is no doubt that this mode realizes the transformation of the enterprise architecture, and takes into account the needs of the business development. It is also easier to be accepted for the smaller impact to the enterprise. Unfortunately, because of the longer time span of this model, when there are contradictory in a single system level between business project and enterprise architecture transformation, people tend to choose development business project first, which result in reducing the effects of the overall architecture governance.

The successful transformation of the enterprise architecture is the core of implementation of EA method, but not the end point. Enterprise architecture is not static. It will change with the adjustment of the macroscopic strategy and microscopic business architecture constantly. Therefore, the maintenance and evolution of the architecture is a continuous process. Architecture management system is a good way to do this long-term work to show and maintain architecture. Once the architectures established, they should keep stability for a period of time. In other words, the architecture designs should be able to meet the needs of business development for a certain period of time. However, the development of the business is not dependent on the will. When business development is beyond the scope of IT architecture, architecture exception is a stopgap solution. Only when the architecture exception accumulate to a certain degree that the architecture cannot meet the needs of enterprise development, a new round evolution of the architecture should be taken into consideration. Generally this evolution cycle should be 3-5 years.

4. Tools and Development Support Systems based on Enterprise Architecture

A. Development tools based on EA

Different with ordinary software development, we need architecture management system, metadata management system and technical design specification, to maintain and query structured or semi-structured enterprise architecture.

Architecture management system is major to management application function, application layer location and its subsystem in application architecture management application relationship according to the relevant rules of the application architecture constraints.

Metadata management system is responsible for the management of various data structures and their corresponding information, including the conceptual data model defined in data architecture and all kinds of further-subdivided Meta data, which define the basic information units and their logical definition that need to be addressed from enterprise level view. Metadata management system also manages the data structure of the various information systems by a clear definition of the physical data for each field, the life cycle of data and the physical definition requirements. On the other hand, metadata management system establishes the correspondence relationship between each data field and the Meta data, accomplishes data architecture mapping from the logical form to the physical form. The data documents and functional interface transmitted among the application systems can also be defined as a form of file management and interface management in metadata management, and fulfill the enterprise data stream management and functional flow management.

Nevertheless, there are still various types of constraints for a large number of enterprise architecture, especially the constraints of technical architecture existing in a variety of technical specification in the unstructured form.

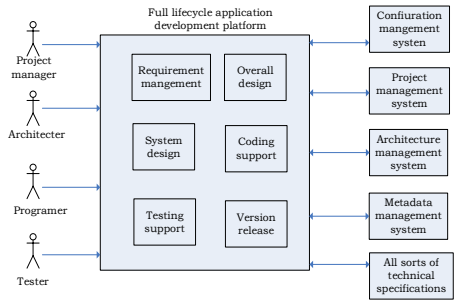

Fig.4 Full lifecycle application development support framework

B. Development support framework based on EA

Fig.4 shows a full lifecycle application development support framework based on enterprise architecture. (1) According to the traditional waterfall software development process, the requirements are decomposed into the structured entry information by requirement management tools in requirements analysis phase, for the positive requirement decomposition and the reverse kinship retrospective. (2) In the stage of the overall program design, the system with automotive linkage to architecture management system and metadata management system by the corresponding design support modules, which inherits the architectural constraints can speed up the design specification and design efficiency; At the same time those against the architectural principles of program design can be marked automatically, for the further analysis and discussion in the peer review process. (3) In the system design stage, the key elements of relevant design, such as the data structure design, component design can be introduced into the metadata management system automatically through appropriate system design support modules, to ensure integrity of key elements of system design and consistency of the architecture management. (4) In the coding and testing stage, various application development platforms can be linked to the configuration management system (source code management) and the defect management system through the framework. (5) In the version release phase, automatic version management, package and version

delivery can be done through the linkage to the configuration management.

5. Practical Effect and Analysis

In order to verify the development method based on Enterprise Architecture, a large-scale promotion of the method has been taken in the software development department of a certain large enterprise since 2008. Comparison of the relevant indicators between those before the implementation of the method (2004-2008) and those after implementation of the method (2008-2012) is as shown in Fig.5.

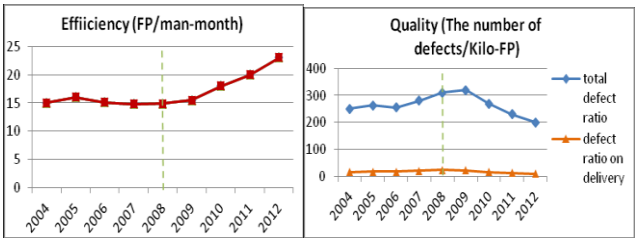

Fig.5 Efficiency and Quality Metrics

It can be found from the above comparative analyses that after the implementation of the method between 2008 and 2012, the overall scale of versions (in terms of FP, Function Point) keep a higher growth rate in comparison with the scale of business. Compared with the past, the development efficiency in 2008 shows a certain degree of reduction, which is due to more time overhead in the process of architecture transformation for the development teams' discussion and analysis of architecture. But after 2010, the development efficiency remains a stable increase. It presents an accelerated state of growth with the gradual development of supporting frameworks (the architecture management system and so on), component library and model library.

On the other hand, the quality of version has promoted significantly because of the involvement of the new method. Relatively speaking, the total defect rate of the version shows a more obvious decrease. Because the enterprise architecture-based product line can receive the long-term accumulation in a certain technical and business area with the implementation of various components, the quality of the development process has significantly improved. Owing to the adjustment of the organizational structure of the product line, the professional integration testing and system testing teams can focus more on business understanding and case accumulation in particular areas, which brings about the significant improvement of defects picked rate of versions. It lays the foundation for the improvement of quality of the delivered version.

6. Further Research

Based on practices in the software development sector of a certain large enterprise, this paper promotes a new component-based development method combined with Enterprise Architecture, implements the whole process including architecture design, architecture transformation, product line organization and full life cycle development framework. Besides, it presents a set of mature and available development methods and institutional system, and builds a development support system based on the enterprise architecture.

In the enterprise, the software development department composed by dozens of development teams that locate in a number of cities and completes hundreds of projects every year. Because of using the component-based software development method combined with enterprise architecture, all these development teams work together around the core enterprise architecture. At the same time by way of a unified development support system, they can maximally extract reusable assets and components, achieve a highly collaborative cross-region inter-department project management, and get good results.

It should be pointed out that the enterprise architecture is a dynamic process of development. How to maintain the stability of the overall enterprise information systems in the continuous evolution of the architecture, and promote the deepening of component-based technology needs for the further research and practice.

Another challenge is the innovation of software development model. In particular, more and more people recognize the agile methods in recent years, due to its short cycle and iteration which satisfies the demand of business development. How to improve the existing mature enterprise architecture based on the waterfall model to meet the needs of agile development is a very promising applied research topic.

Acknowledgment

Thanks for support from the major national science and technology special projects (2010ZX03004-003-03) and the fundamental research funds for the central universities (3105005, 111087).

References

[1] Mei Hong, Shen Jun-Rong. Progress of Research on Software Architecture. Journal of Software, pp.1257-1275, Vol.17, No6, JUNE.2006.

[2] Wang Xiao-yan, Liu Shu-fen, Zhang Jun. A Framework baesd on Domain Model and Component Composition. Acta Electronica Sinica, pp540-545, Vol37, No 3, 2009.

[3] J.A. Zachman. A framework for information systems architecture. IBM system journal Vol. 26, No 3, 1987.

[4] The Open Group, "Togaf enterprise edition version 9, "2009.[Online], http://www.opengroup.org/architecture/togaf9-doc/arch/index.html

[5] Zeng Sen, Fan Yu-shun. Service-oriented enterprise architecture. Application Research of Computers, p541-545,Vol 25, No 2, FEB 2008.

[6] Clements P, Northrop L. A Framework for Software Product Line Practice. Version4.2 http://www.sei.cmu,edu/productlines/framework, 2004.