International Journal of Hybrid Information Technology
Vol.8, No.3 (2015), pp.375-386
http://dx.doi.org/10.14257/ijhit.2015.8.3.34

The Research of College Sports Information Management System Based on Component and the Application in Student's Management Work

Liang li[1,*] and Ligang Tian[2]

[1]Collegeof information science and technology, Agricultural University of Hebei
Baoding, Hebei, China
[2] Sports Work Department, Agricultural University of Hebei
Baoding, Hebei, China
xxxytw@sina.com

Abstract

Software reuse is the process of building software application that makes use of formerly developed software components. Software reuse has the potential to shorten the delivery times, improve the quality and reduce the development costs. Software reuse which is based on component is the effective solution to solve these problems. With the rapid development of the computer technology, the college sports information management is widely applied in college sports education. And the sports management system is concerned by the physical educator, especially the student's management work. The research of the sports information management system lags behind relatively at home. In this paper, we propose a model of sports information management system based on the component. And we apply this sports information management system in student's management work. The application of this system improves the work efficiency and saves a lot of manpower and material resources. The system effectively regulates the cooperation between various departments, provides an important guarantee for the integrity of student information and promotes the informationization construction of school. This model is particularly instructive and relevant to the development of sports information management system.

Keywords: *Software reuse, College sports student information management system, Student's management work*

1. Introduction

With the rapid increase of the software requirements, the scale and complexity of the software increase continuously. The traditional software development mode faces the hitherto unknown challenge. Therefore, scholars begin to explore new software development technology in order to adapt to the requirement of the software development. The software reuse is an effective way to solve the software crisis. And the software component technology plays an important role in the software reuse. The software reuse is a basic characteristic of the mature engineering field, such as the civil engineering, the chemical engineering and the computer hardware engineering. In addition, software reuse avoids the repeated design in the system development and reduces the development cost greatly. It also improves the production efficiency and the product quality. Similarly, the software reuse is an indispensable route which makes the software engineering mature. And it provides a feasible way to solve the software crisis.

The software development technology based on the soft-components is a solution which can avoids the repeat work in the software development. Software development technology is also an application of the software reuse in practice. The starting point of the software development technology is based on the existing work. This mode is not the "from scratch" any more. Now, the development technology makes the software architecture as the blueprint, takes the components as the assembly and supports the packaged development. Thereby, software reuse not only achieves the minimum coding but also reduces the workload of testing and maintenance. In addition, the software reuse is one of the effective way to eliminate the reduplicative work. And it also avoids the possible errors in redevelopment, improves the efficiency of the software production and shortens the delivery time of the software production.

The software reuse is the development of new software as sets by reusing and extending existing software assets, such as libraries, frameworks, and components. In the real life, the software maintenance cost is increasing continuously [1, 2]. We use the distributed object technology to alleviate the software crisis. The distributed object technology provides strong support for the software reuse [3, 4]. Potential benefits of the software reuse include the reducing development time and costs. It also shortens time-to-market and improves the software quality. Reusable software assets are often prefabricated blocks of code that are assembled to provide new software solutions. They range from source code, development plans, requirement analysis, and design documents to test cases. The development and integration of reusable software assets have been recommended to solve the problems due to budget overrun, late delivery, and project failure [5]. Proponents of reusability believe that the approach is an effective strategy for improving the efficiency and quality of the software systems. And the approach reduces the time to market [6, 7]. Companies have invested millions of dollars to populate libraries of reusable software assets and trained employees to generate and use these assets. Prior researches on reusable assets have explored different forms of assets [8-10] and barriers to the development [11]. Past experiences have shown that the non-technical issues are difficult to resolve both the organizational and individual levels [12].

The most advanced reviews which are based on two assumptions do not reflect the application of the software reuse very well. Firstly, many scholars assume that the cost of reusing a software asset depends on its size [13-15]. However, the size is not always a good predictor for this cost. For example, ''off-the-shelf'' components are often adjusted to the context in which they are used through parameterization [16, 17]. The cost of reusing will primarily be determined by the number of parameters that need to be configured. The number of parameters depends on the commonality among the applications in the target domain, rather than its size. Secondly, existing reuse economic models develop the asset for one-time use by the ''Relative Cost of Writing for Reuse'' [18, 19]. Because of reusability being added to every part of these assets, this approach reflects the case when reusable assets are developed from scratch. In practice, however, the software reuse is often applied recursively, i.e. applications are developed by reusing and extending one or more existing software assets that have been developed with reuse as well [20-22].

In this paper, we analyze a software reuse technology and the software development method comprehensively. According to the domain analysis results and the theory of the knowledge system architecture, we propose a domain software structure which is based on a college sports information management system and we apply this system in student's management work. The architecture can obtain the functional model of the main components and the hierarchical model of the system through detailed field analysis. This system provides the complete business operation process for school student's work management teacher and promotes the informationization construction of school. The construction of this paper is as follows. Firstly, this paper introduces the related background. The second part is the software reuse. In this part, we introduce the basic acknowledge of software reuse. The third part is the college sports information management system based on the component. In this part, we propose the college sports information management system based on the component. The fourth part is the component management. In this part, we propose the storage mode of college sports information management system. The last part is the conclusion

2. Software Reuse

2.1 Composition and Variation

A new software asset is created by reusing and extending one or more existing software assets when we apply the software reuse. Generally speaking, there are two mechanisms exist to achieve the software reuse. They are composition and variation. Composition refers to the integration of two or more existing software assets that have been developed independently from each other. And variation refers to the ability to adjust a software asset to the context in which it is used

When we apply variation, the behavior of a reusable asset is configured to fit the requirements at hand. Each variation point is associated with many alternative choices which are called variants. A schematic overview of a reusable asset Z (the central rectangle) is shown as Figure 1. And its variation points are numbered A, B, and C.

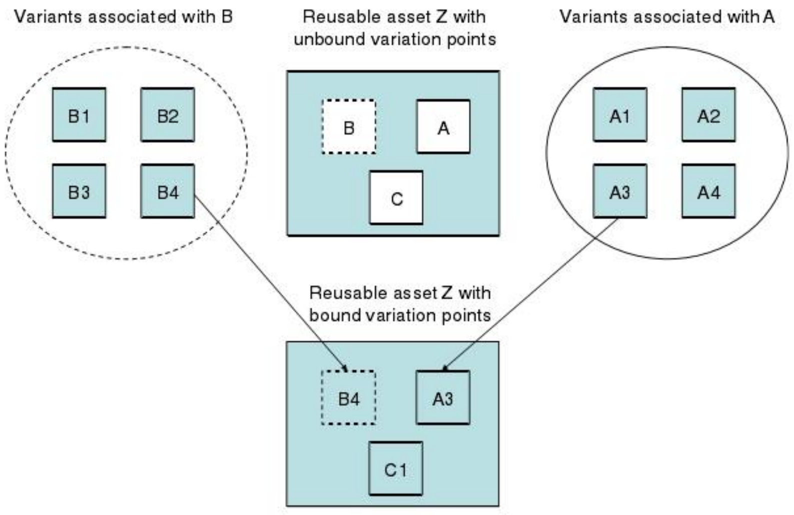

Variants associated with B
Reusable asset Z with unbound variation points
Variants associated with A
B1
B2
B3
B4
B
A
C
A1
A2
A3
A4
Reusable asset Z with bound variation points
B4
A3
C1

Figure 1. A reusable Asset with Variation Points, Numbered A, B, and C, and Associated Variants

2.2 Three Different Mechanisms of the Software Reuse

As we know, there are three different mechanisms of the software reuse. They are white-box variation, black-box variation and composition. White-box variation is used to refer to the process of the developing one or more new variants. Black-box variation is used to refer to the configuration of a reusable asset without developing new variants. Composition requires software developers to understand the internal functioning of this product family. Three different mechanisms of the software reuse are shown as Figure 2.

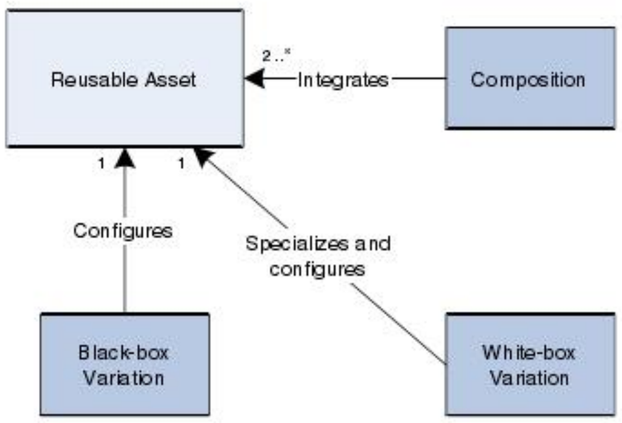


Figure 2. Three Different Mechanisms of Software Reuse

2.3 Software Reuse System

If the design of the software reuse system is based on the component, the process is similar to the general software engineering design. The mainly design procedures are as follows.
1. The demand analysis. It mainly combines with the special requirements of the application system to analyze the results of the demand.
2. The functional analysis. This analysis describes the system function.
3. The detailed design. Programmers build the algorithm design of the software.
4. The coding. It constructs the executable code.
5. The operation maintenance. Maintenance is performed at the functional level.
6. The software reuse and understanding. They can be used as the reusable component storage.

2.4 The Hierarchical System Structure of Component

The system structure of the component is layered architecture. The component is usually divided into three layers. They are the basic component layer, the domain general component layer and the domain specific component layer. It is described as follows.

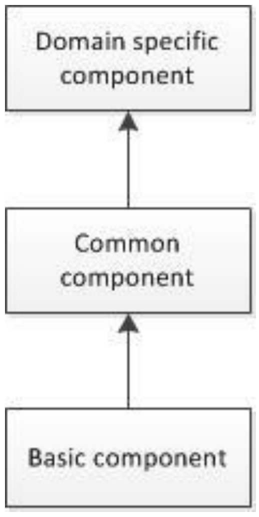


Figure 3. Component Hierarchical System

The bottom is a foundation component. It is the component which is used in the integrated support environment and the operational environment. In this paper, it refers to the tool mean, data window, buttons and the other components. The middle layer is a field of universal component such as the report component of MIS and the query component. The top is domain specific component. The construction is lower level, the function is much simpler and its flexibility is higher. The construction is higher level, the function is stronger and its flexibility is poorer.

3. College Sports Information Management System Based on Component

In this section, we construct the college sports management system by demand analysis. We put forward college sports information management system based on component. It contains 7 diagrams. They are superior file management, teacher management, students management, teaching management, competition management, equipment management and sports test management. The structure charts are as shown.

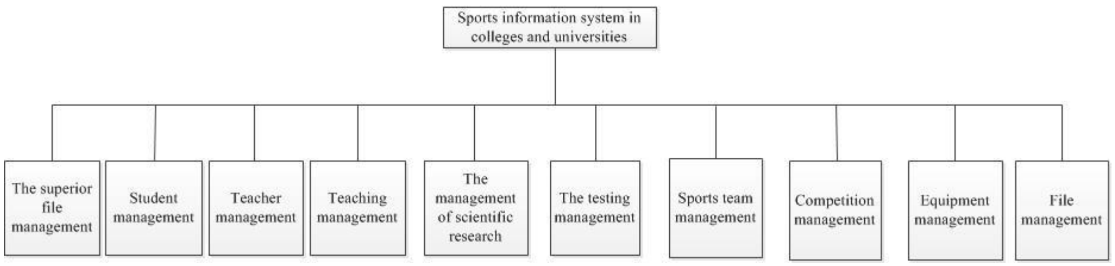


Figure 4. Sports Information System in Colleges

The college sports information system includes the superior file management, Teacher management, Students management, Teaching management, Competition management, Equipment management, and sports test management.

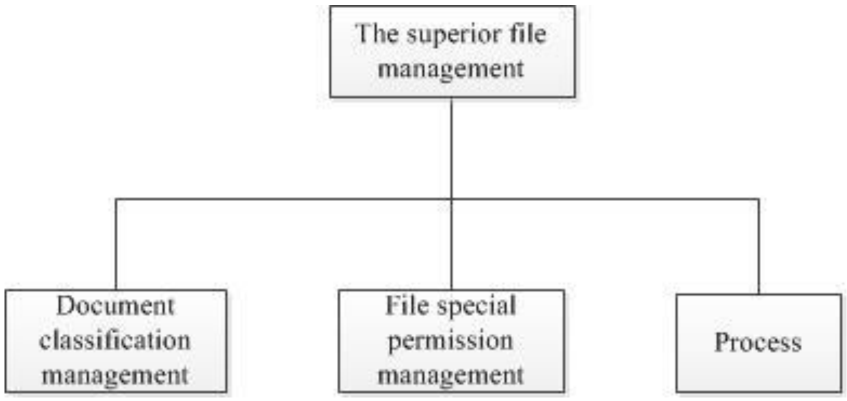

The superior file management
Document classification management
File special permission management
Process

Figure 5. The superior File Management

The superior file management includes document classification management, file special permission management and the process.

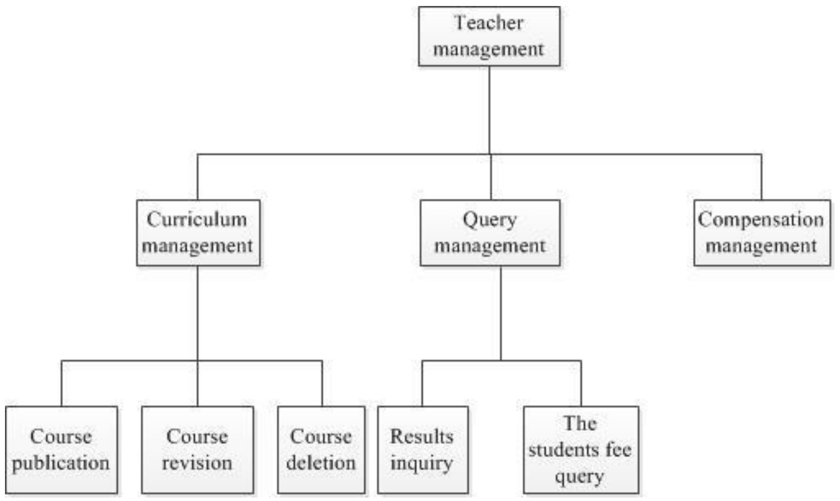


Figure 6. Teacher Management

Teacher management includes the curriculum management, the query management and the compensation management. The goal of the curriculum management is to publish the course information, revise the course information and delete the course information. The function of the query management is querying the results and the student fee. The compensation management is to manage the remuneration of teachers.

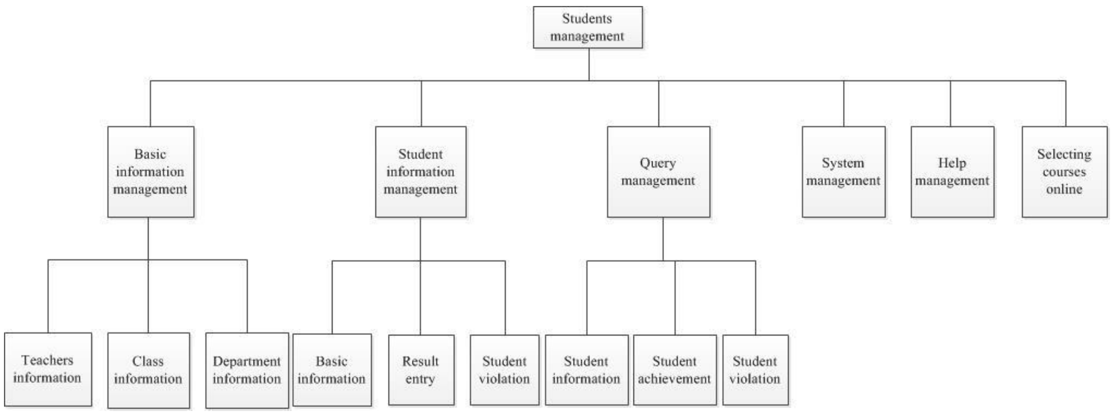


Figure 7. Students Management

Student management includes the basic information management, student information management, query management, system management, help management and selection course online. The role of the basic information management is to manage the teacher information, class information and department information. Student information management includes student basic information, result entry and student violation information. The function of the query management is to query the student information, student achievement and the student violation information. System management is to manage the system. If there is any question, you can ask the help management. Selecting courses online is a system to select the courses online.

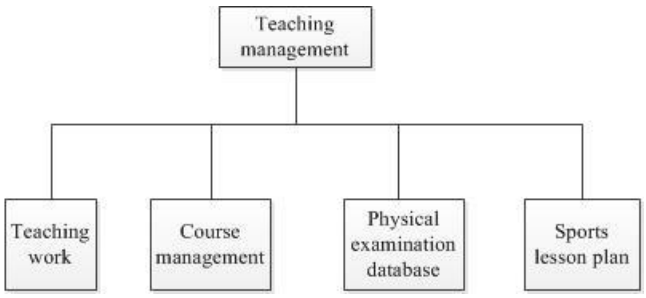


Figure 8. Teaching Management

Teaching management contains the teaching work, course management, physical examination database and sports classroom. The function of the teaching work is to draw up a teaching plan, write the syllabus and so on. The role of the course management is to check the course plan, adjust the course schedule and so on. Physical examination database includes many physical theoretical test questions. Sports classroom embraces the teaching plans.

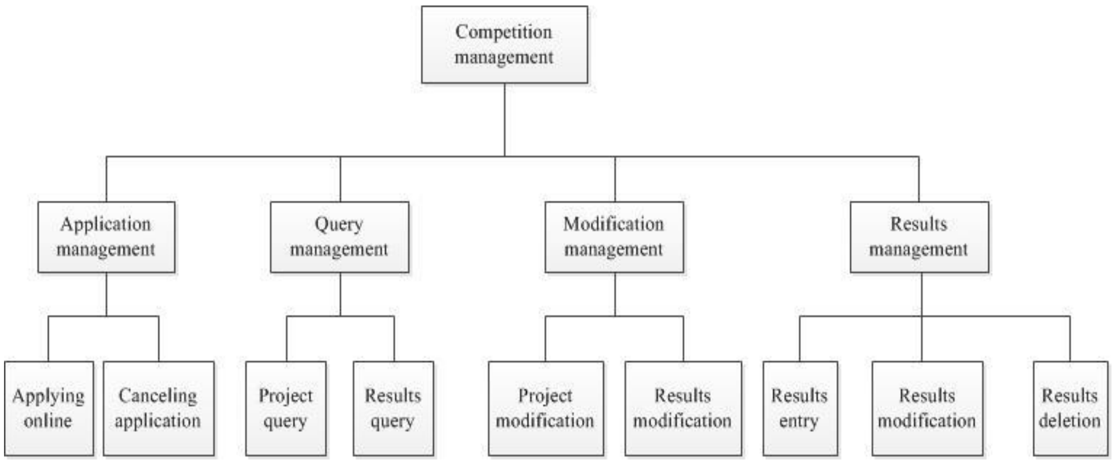


Figure 9. Competition Management

Competition management covers application management, query management, modification management and results management. If you want to apply the competition or quit the competition, you can entry application management. If you want to query the project and the results, you can entry query management. Administrators entry the modification management to manage the project information and the results information.

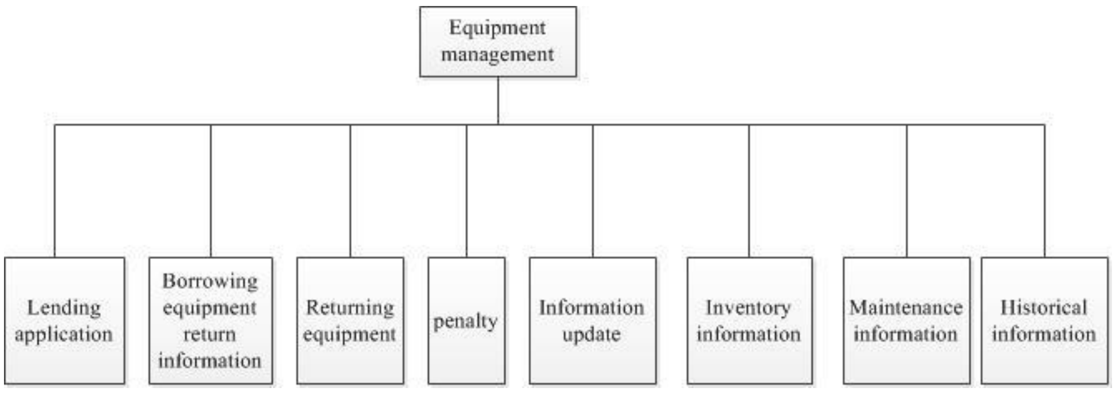


Figure 10. Equipment Management

Equipment management includes the lending application, borrowing equipment return information, returning equipment information, penalty information, information update, inventory information, maintenance information and historical information.

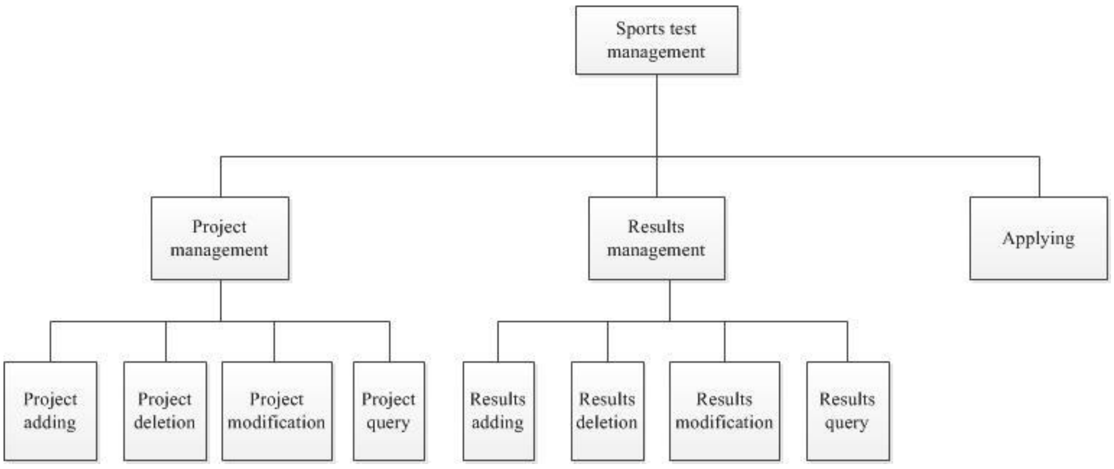


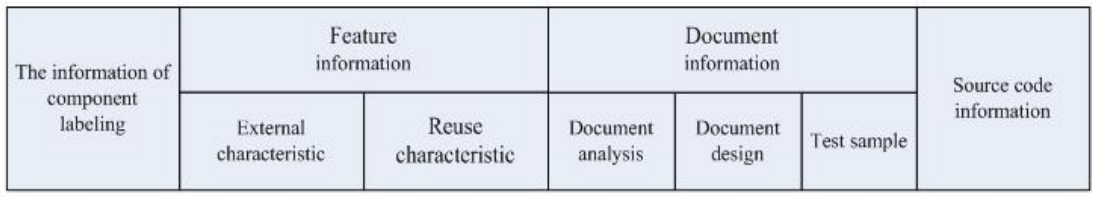
Figure 11. Sports Test Management

Sports test management covers project management, results management and applying. Project management is to manage the project for adding, deleting, modification and querying. Results management is to manage the project for adding, deleting, modification and querying. Applying is to apply for sport tests.

4. The Component Management

4.1The Storage of Component

Any simple memory system cannot store and manage the component flexibly. Thereby, we use the relational data base to store the labeling information of the component and use the document system to store the material part of the object. We build the map link between the two parts. The Storage structure of the component is shown as follows.



The information of component labeling	Feature information		Document information			Source code information
	External characteristic	Reuse characteristic	Document analysis	Document design	Test sample	

Figure 12. The Storage Structure of the Component

The information of the component label is to show the classified information. The feature information is to express the related information and the inherent information of the component management. So we retrieve and match the component quickly. We describe the component from the external characteristic and the reuse characteristics. The external characteristic includes component type, number of reuse, key word, abstract (describing the function of the component, reuse information and so on). The reuse characteristic includes the attribute of the component related classes, IDL document, component platform, development language and so on.

4.2 Storage Style of Component

The component base system is based on database management system. But the storage format of the source code for component is different from the storage format for database.

Therefore, the component is not suitable to store in database. We storage the text information and the source code in a special system. In database, we only store the storage path information of the map file. Storage style of component is shown as follows.

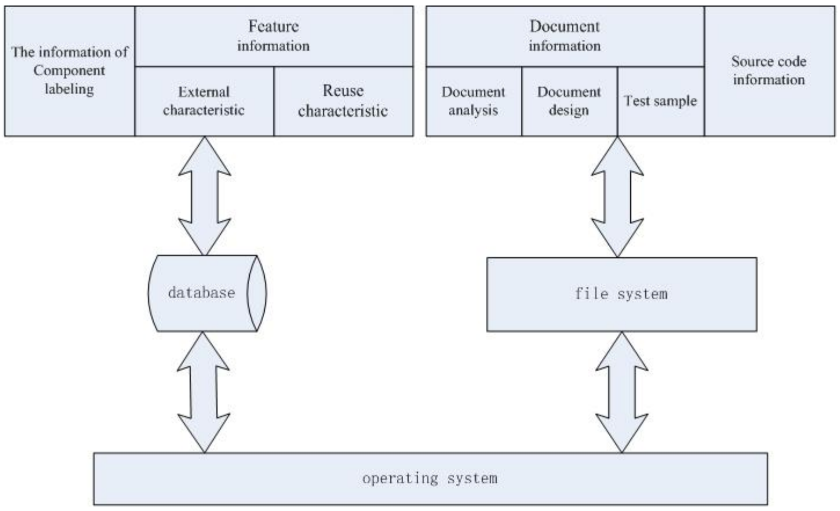


Figure 13. Storage Style of Component

4.2 Component Library Function System

The component library system is used to store and manage the reusable component. The functions of this system are storing, adding, deleting, modifying and retrieve the component. And we add the functions of component reuse record and user feedback. Component library system function is shown as follows.

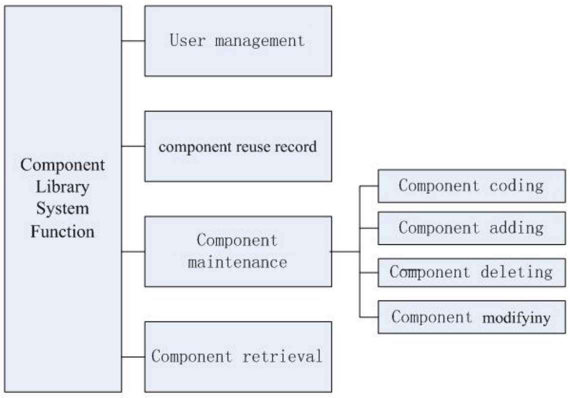


Figure 14. Component Library Function System

4.3. Component Library System

The main problem that the component library solved is how to establish the component library. Namely, how we lead the component into the component library system. The component library system is shown as follows.

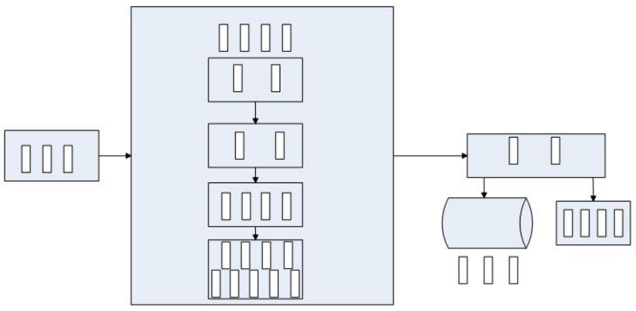


Figure 15. Component Library System

There are three parts of the component entering library. They are component test, data preparing and entering library. In the part of the component test, we test whether there is quality problem in the component. In the stage of the data preparing, we classify the component based on component classification method. Then we catalog the component to make sure the sequence number and version number of the component. In the entering library part, we ensure the storage path of the text and source code of the component. The component entering library is finished after the related data is prepared.

5. Conclusion

In this paper, we not only introduce the software reuse but also establish a college sports information management system. Though this system, the teachers can master the data of students timely and conveniently. This application improves the work efficiency and reduces the education cost. At last, we discuss the component management in detail. We have done a lot of work: (1) we introduce two ways of software reuse: composition and variation. (2) We list three different mechanisms of software reuse, software reuse system and the hierarchical system structure of component. (3) We establish a college sports information management system based on component. (4) We apply this system in student’s management work. (5) We discuss the realization of component management. There are outstanding advantages and strong vitality in the software reuse and reusable component technology. Thereby, this college sports information management system has a broad application space.

References

[1] W. Frakcs and P. Gandel, “Representing Reusable Software”, Information and Software Technology, vol. 32, no. 10, **(1990),** pp. 654-664.

[2] W. B. Frakes and B. A. Nejmeh, “An Information System for Software Reuse”, In: Software, Reuse: Emerging Technology, IEEECS Press, **(1990)**, pp. 142-151.

[3] M. M. Rafique, A. R. Butt and E. Tilevich, “Reusable software components for accelerator-based clusters”, Journal of Systems and Software, vol. 84, no. 7, **(2011),** pp. 1071-1081.

[4] Z.-W. Hong, J.-M. Lin, H. C. Jiau, G.-M. Fang and C. W. Chiou, “Encapsulating windows-based software applications into reusable components with design patterns”, Information and Software Technology, vol. 48, no. 7, **(2006),** pp. 619-629.

[5] S.-C. Chou and Y.-C. Chen, “Retrieving reusable components with variation points from software product lines”, Information Processing Letters, vol. 99, no. 3, **(2006),** pp. 106-110.

[6] T. Ravichandran and M. A. Rothenberger, “Software reuse strategies and component markets”, Communications of the ACM, vol. 46, no. 8, **(2003),** pp. 109–114.

[7] P. Jing, J. Qiyun and H. Dan, “A reusable design of geophysical data acquisition software”, Procedia Engineering, vol. 16, **(2011),** pp. 239-244.

[8] C. Ziemke, T. Kuwahara and I. Kossev, “An integrated development framework for rapid development of platform-independent and reusable satellite on-board software”, Acta Astronautica, vol. 69, **(2011),** pp. 583-594.

[9] Z. Tang, X. Zhang and L. Huang, “Design and Implementation of Reusable Components Using PowerBuilder”, Procedia Engineering, vol. 29, **(2012),** pp. 584-588.

[10] V. Subedha and S. Sridhar, "Optimization of Component Extraction for Reusable Software Components in Context Level – A Systematic Approach", Procedia Engineering, vol. 38, **(2012),** pp. 561-571.
[11] C. Fournier, C. Pradal, M. Chelle, F. Boudon, G. Louarn, C. Robert, D. Combes, T. Cokelaer, J. Bertheloot, K. Ma, S. Saint-Jean, A. Verdenal, A. Escobar-Gutièrrez, B. Andrieu and C. Godin, "Sharing efforts for modeling plant systems: from publications to reusable software components", Comparative Biochemistry and Physiology Part A: Molecular & Integrative Physiology, vol. 23, no. 2, **(2009),** pp. S222.
[12] E. Y. Nakagawa, F. C. Ferrari, M. M. F. Sasaki and J. C. Maldonado, "An aspect-oriented reference architecture for Software Engineering Environments", Journal of Systems and Software, vol. 84, no. 10, **(2011),** pp. 1670-1684
[13] G. Bockle, P. Clements, J. D. McGregor and K. Schmid, "Calculating ROI for software product lines", IEEE Software, vol. 21, **(2004),** pp. 23–31.
[14] B. Boehm, A. W. Brown, R. Madachy and Y. Yang, "A software product line life cycle cost estimation model", in: Proceedings of the 2004International Symposium on Empirical Software Engineering, IEEE Computer Society, Washington, DC, **(2004),** pp. 156–164.
[15] C. Ayala, O. Hauge, R. Conradi, X. Franch and J. Li, "Selection of third party software in Off-The-Shelf-based software development—An interview study with industrial practitioners", Journal of Systems and Software, vol. 84, no. 4, **(2011),** pp. 620-637.
[16] V. Mellarkod, R. Appan, D. R. Jones and K. Sherif, "A multi-level analysis of factors affecting software developers' intention to reuse software assets: An empirical investigation", Information & Management, vol. 44, no. 7, **(2007),** pp. 613-625.
[17] M. R. J. Qureshi and S. A. Hussain, "A reusable software component-based development process model", Advances in Engineering Software, vol. 39, no. 2, **(2008),** pp. 88-94.
[18] O. Hauge, C. Ayala and R. Conradi, "Adoption of open source software in software-intensive organizations – A systematic literature review", Information and Software Technology, vol. 52, no. 11, **(2010),** pp. 1133-11154.
[19] O. Mazhelis, P. Tyrväinen and L. Frank, "Vertical software industry evolution: The impact of software costs and limited customer base", Information and Software Technology, vol. 55, no. 4, **(2013),** pp. 690-698.
[20] I. S. Souza, G. S. da Silva Gomes, P. A. da Mota Silveira Neto, I. do Carmo Machado, E. S. de Almeida and S. R. de Lemos Meira, "Evidence of software inspection on feature specification for software product lines", Journal of Systems and Software, vol. 86, no. 5, **(2013),** pp. 1172-1190.
[21] M. Sarrab and O. M. H. Rehman, "Empirical study of open source software selection for adoption, based on software quality characteristics", Advances in Engineering Software, vol. 69, **(2014),** pp. 1-11.
[22] R. V. Ommering, "Software reuse in product populations", IEEE Transactions on Software Engineering, vol. 31, **(2005),** pp. 537–550.