Design and Implementation of RAS-Based Reusable Asset Management Tool

Ze-Sheng Wang, Ru-Zhi Xu, Kang-Kang Zhang, Da-Li Feng
School of Computer Engineering, Shandong University of Finance, Jinan, China, 250014
ksuperboy@126.com

Abstract

In order to manage asset library effectively, this paper represents an approach of RAS-based asset management. The paper firstly extends the Default Component Profile of RAS to improve software reuse, and then present tool's architecture based on the extended RAS, which is divided into main system and assistant system according to the function of every part of the tool. Finally, the tool is implemented to show that this solution is feasible and effective.

1. Introduction

Component-Based Software Development (CBSD), as a means of software reuse, is a feasible way to improve software productivity and quality [1, 2]. With software enterprises' turning to CBSD, how to deepen enterprises' CBSD process is one focus taken seriously by software enterprises.

A few components can be managed by hand. But it will become difficult to manage thousands of components manually. CBSD process includes requirement analysis, retrieving components, getting components, and integration of applications software. Retrieving and getting components is a process of communicating with domain component library (asset library).

Software reuse includes two processes: 1) developing reusable components; and 2) assembling software systems from software components. As viewed from Domain and Application Engineering, domain component library associates Domain Engineering with Application Engineering.

Actual REBOOT [3] and ALOAF [4] is designed for common library and not suitable for domain component library. And at the same time, the management of domain component library is different from it of common library.

The objectives of this paper are to design and implement an RAS-based asset management tool which manages the enterprises' domain component library.

The reminder of the paper is organized as follows. Section 2 introduces the RAS and shows how to extend the RAS. Section 3 represents the design and implementation of asset management tool. Section 4 concludes the paper. And section 5 is acknowledgement.

2. Extending RAS

The description of components is the foundation of the software reuse. There exist some component models at present, such as 3C [5] and REBOOT [3] which adopt facet-based classification [6]. But the facet classification is not suitable for building domain component library.

2.1. Reusable asset specification

Reusable asset specification (RAS) [7] is an Object Management Group standard to describe and package asset. RAS is described in two major categories, Core RAS and Profiles. Core RAS represents the fundamental elements of asset specification. Profiles describe extensions to those fundamental elements. The Core RAS is not instantiated therefore an asset must be of a particular profile. A profile may extend Core RAS or may extend another profile. This Default profile is a realization of the Core RAS. The Default Component Profile and Default Web Service Profile derive from the RAS Default Profile.

Core RAS defines four major sections to an asset including the Classification section, Solution section, Usage Section, and Related-Assets section. This paper extends Default Component Profile in order to improve software reuse.

978-0-7695-3112-0/08 $25.00 © 2008 IEEE
DOI 10.1109/ICICSE.2008.63

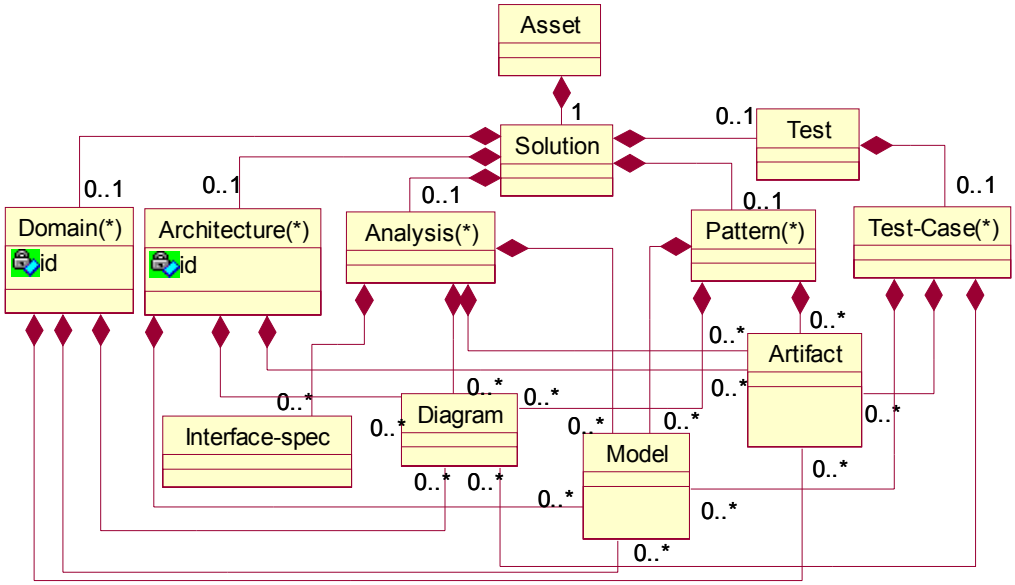

Figure 1. Extending the Solution section

2.2. Extending RAS

As shown in Figure 1, the Solution section is mainly extended and the classes marked by asterisk are newly added.

The newly added classes, Domain, Architecture, Analysis, Pattern and Test Case, organize special kinds of Artifacts that improve browsing and navigation of the asset. The reason of extending Solution section is based on the following issues:

- Representing the mapping relationship of Domain Model, Architecture, and Component

Domain Models, Architectures, and Components are the main reusable assets. Domain Model is a model which describes the common requirement among the systems in a domain [8]. Architecture is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them [9]. Domain Model, Architecture, and Components are created from domain analysis, and Domain model is mapped to Architecture [10].

The relationship of the main reusable assets, Domain Model, Architecture and Components, is represented through the newly added classes, Domain and Architecture, and is managed automatically by asset management tool. In this context, the associations among components can be established automatically, and then the chance that the main assets are reused is improved greatly.

- Separating more reusable artifacts

In different environment, Analysis Models, Patterns and Test Cases also can be reused. So the classes, Analysis, Patten and Test Case are added to separate more reusable artifacts from the reusable assets.

In addition to extending the Solution section, the Descriptor Group class of the Classification section is extended using facet classification. The newly added classes are Functional Type, Programming Language, Operation System, Encapsulation Style, Container Compatibility, and Internationalization. Thus, the approach of retrieving components based on faceted classification [11] can be used.

3. Design and implementation of asset management tool

3.1. Design of asset management tool

As shown in Figure 2, the section surrounded by broken lines is the Asset Management Tool (AMT) which is divided into Main System (MS) and Assistant System (AS). The MS is composed of assets submission, retrieving assets and usage of assets. The AS is composed of user and privilege management, information statistics, project information management and decision-support system. The supporting function to CBSD is provided by the MS section of the AMT.

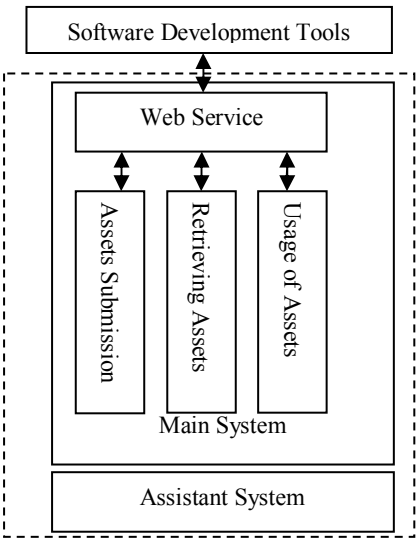

Figure 2. Asset management tool

3.1.1. Main system.

- Assets Submission

Assets are described by extended RAS, and then are compressed into .RAS files using the ZIP algorithm. When the needed assets are not found in the domain library, AMT can communicate with common component library through internet. Now, some common component repositories provide Web Service through which AMT can retrieve components easily, such as Component Source [12].

This functional module includes uploading assets, uploading artifacts, assets information and association of assets, and is mainly used by assets providers.

- Retrieving Assets

It is important to retrieve needed assets quickly from component library. The Classification section is used to retrieve assets. Users can get more needed assets using assets recommendation (see 3.1.2).

This functional module includes retrieving assets, advanced retrieving, retrieving profile and classification retrieving, and is mainly used by software developers.

- Usage of Assets

The Usage section describes the activities to be performed for applying or using the asset. This section is described with a lightweight activity or workflow model. For example, a <variability-point> is a conceptual spot in an artifact that is expected to be altered by the asset consumer. It describes where and what in the artifact can be modified.

This functional module includes browsing detail of assets, browsing the usage of assets and downloading assets and artifacts, and is mainly software developers.

3.1.2. Assistant system.

- User and Privilege Management

Different users should be given different privilege. For example, asset providers can just submit asset, while administrators can add or delete users.

This functional module includes adding users, deleting users and privilege maintenance, and is mainly used by administrators.

- Information Statistics

The main function is to find associations among assets using association rules. Thus, the assets which are often used together by users can be found, and then the relevant assets are recommended to users when users are browsing or downloading assets.

This functional module includes statistics of the usage of assets, analysis of association of assets, browsed assets' rank and downloaded assets' rank, and is mainly used by administrators.

- Project Information Management

Because software is developed under a project, it is indispensable to manage project information.

This functional module includes projects submission, project management information and maintenance of the association of projects and assets, and is mainly used by administrators.

- Decision-Support System

Reuse and economy profit statistics both provide support to decision-makers. Thus, the decision-makers will be determined to which components should be bought, and which components should be developed.

This functional module includes reuse statistics and economy profits statistics, and is mainly used by decision-makers.

3.2. Implementation of asset management tool

Based on the study of the paper, we develop the RAS-based Asset Management Tool system. The system is developed using Java programming language. The system can manage the whole assets' life cycle, and support CBSD process well, and facilitate assets management. Figure 3 shows the interface that assets are described using extended RAS, and then are compressed to .RAS files using ZIP algorithm.

The main system is encapsulated into Web Service, and then can be integrated into software development tools. Thus, through software developer tools, software developers can submit or download assets, and can browse the usage information of assets.

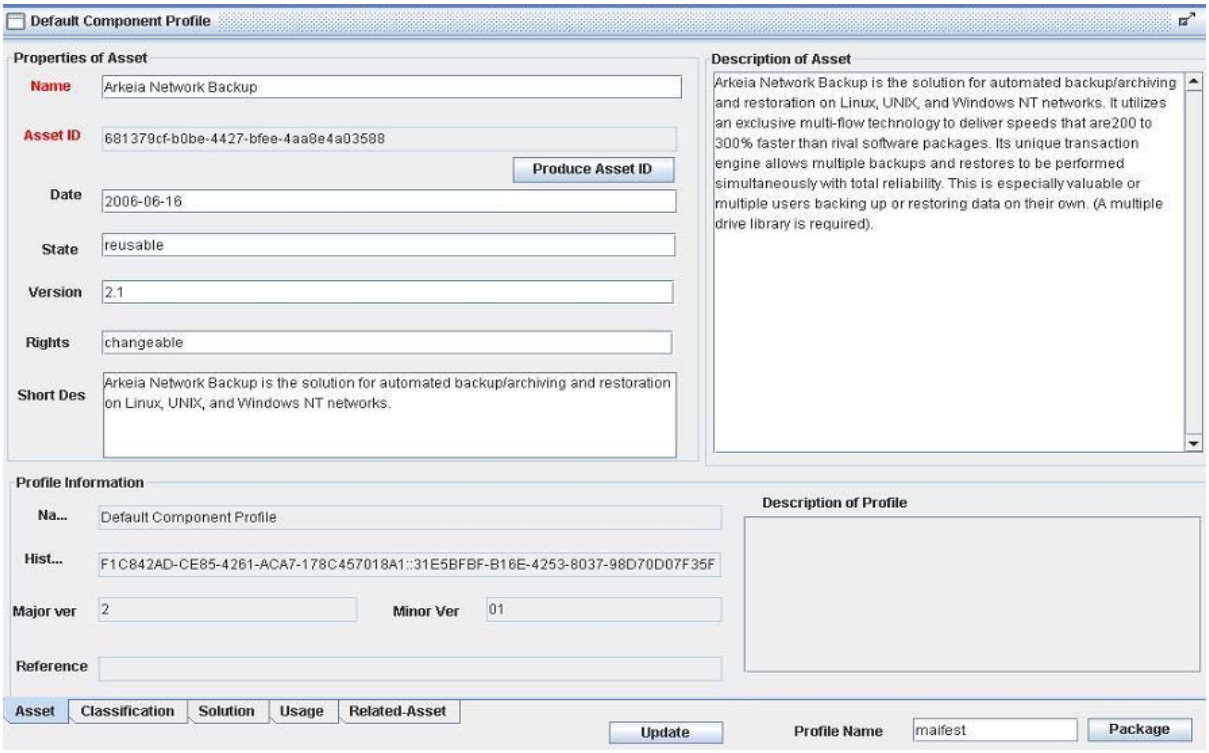

Figure 3. Interface of describing assets

4. Conclusion

The enterprises' asset library can be managed effectively by the Asset Management Tool. The main reusable assets, domain model, architecture, and components, can be managed by the tool in the whole life cycle.

5. Acknowledgement

This work is supported by Natural Science Foundation of Shandong Province (A200621) and Science & Technology Bureau of Jinan (061106).

6. References

[1] Ruben Prieto-Diaz, "Status Report: Software Reusability", *IEEE Software*, Vol. 10, No. 3, May 1993, pp. 61-66.

[2] Ru-Zhi Xu, and San-Yuan Zhu, "Research on Matching Algorithm for XML-Based Software Component Query", *Journal of Software*, Science Press, Beijing, China, 2003, Vol. 14(7), pp. 1195-1202.

[3] Morel J M, and Faget J, "The REBOOT Environment [C]", Prieto2Diaz R, FrakesW B. Proceedings of the 2nd International Workshop on Software Reusability Advances in Software, Lucca, IEEE Computer Society Press, 1993, pp. 80-88.

[4] STARS Technical Committee, "Asset Library Open Architecture Framework: Version 1.2", Informal Technology Report, STARS-TC-04041/001/02, August (1992).

[5] Tracz, and Will, "The 3 Cons of Software Reuse", Proceedings of the 4th Workshop on Software Reuse, July 1990.

[6] V. Lucena, "Facet-Based Classification Scheme for Industrial Automation Software Components", Proceedings of Sixth International Workshop on Component-Oriented Programming (WCOP 2001), Budapest, Hungary, 2001

[7] OMG, "Reusable Asset Specification", Available Specification Version 2.2, formal/05-11-02.

[8] James Petro, Michael E. Fotta, and David B. Weisman, "Model-Based Reuse Repository - Concepts and Experience", Proc. Seventh International Workshop on CASE, Toronto, Ontario, Canada, Edited by Hausi A. Muller & Ronald J. Norman, IEEE Computer Society Press, Los Alamitos, California, USA, July 10-14, 1995, pp. 60-69.

[9] Len Bass, Paul Clements, and Rick Kazman, "Software Architecture in Practice. 2/E", Addison-Wesley Professional, 2003.

[10] Jacques Meekel, Thomas B. Hortont, Robert B. Francet, CharlieMellone, and Sajid Dalvi, "From Domain Models to Architecture Frameworks", ACM O-89791 -945 -9/97 JOO05, 1997, pp. 75-85.

[11] WANG Yuan-feng, ZHANG Yong, and REN Hong-min, "Retrieving Components Based on Faceted Classification", *Journal of Software*, Science Press, Beijing, China, 2002, 13(8), pp. 1546~1551.

[12] Component Source, http://www.componentsource.com