Documentation of Software Components for Reuse

Neha Malik
Dronacharya College of Engineering, Gurgaon, India
nehag78@gmail.com

Varsha Shrivastava
Dronacharya College of Engineering, Gurgaon,, India
shrivastava.varsha@gmail.com

***Abstract:* - Component documentation has become a key issue in component trading because it often is the only way of assessing the applicability, reliability and quality of a third-party component, especially for product lines in which the common architecture determines the significant requirements and restrictions for components. Definition of the documentation pattern is not sufficient for the adoption of a new documentation practice. An environment that supports the development of documentation is also required. This paper highlights the difference between various document models discussed in the paper.**

Keywords: Components, Documentation, Framework.

I. INTRODUCTION

Dusinkand Katwijk (1995), in their survey on there use literature, argued that component documentation was a neglected area of research. However, a few models for component documentation have been published. We chose the following ones for more detailed analysis: Sametinger’s (1997), Karlsson’s (1995), and the NATO model (1992). We naturally recognize a wide base of literature concerning the documentation of code components (Frakes &Pole, 1994;Henninger, 1997). However, the emphasis in these is on find in gander trieving components, or they address other narrow and specific areas: e.g., how to document loops.

II. REUSABLE SOFTWARE COMPONENT DOCUMENTATION MODELS

This Paper concentrate on the three models mentioned earlier appropriate for enlightening the state-of-art in general and for fulfilling the purposes of this paper. We analyzed and compared Same tinger’s, Karlsson’s and NATO’s models in light of Or likowski’s and Yates’(1998)genre system frame work. This section attempts to summarize these models and their short comings as a basis for our model to be elaborated.

A. The NATO Model for the Reusable Software Component Documentation (NATO,1992)

The NATO standard for the development of reusable software components (RSC) pursues maximum potential for software reuse. Documentation of are usable component provides a key part of its reuse value, play in gadual role. First of all, it must conform to the needs of the immediate software system under preparation. Secondly, it must give explicit guidance for re-users. Are-user must be able to access quickly the information.

Documentation must comply with accepted standards of the user community, being consistent in organization and in format, and reflecting changes in the code. Documentation should be self-contained and possibly accompanied with their usable component. Finally, documentation should be in machine-read able form and understand able by others.

Reuse library emerges as a special concern. Documentation must support the classification, identification and retrieval of components .A component’s functionality should be easily viewable through abstracted summaries. The dependencies must be explicitly described and there should be classification information.

Also the assessment of RSC should be described. Different kinds of metrics about there usability and quality should be stated as well as known problem sand recommended enhancements. The potential re-user should also be aware of any commercial or legal restrictions, and how to access the component if it is not physically in there use library.

Reuser’s Manual

1. INTRODUCTION
   - purpose of the document
   - overview of the component
2. FUNCTION
   - operation
   - scope
3. INTERFACES
   - RSC specification (identify all externally visible operations)
   - external references and parameters
   - interfaces by class
4. PERFORMANCE
   - assumptions
   - resource requirements
   - exceptions (how the RSC responds to incorrect inputs)
   - test results (any performance measurements)
   - known limitations
5. INSTALLATION
   - how to instantiate the component (e.g., generic parameters)
   - interfaces (enumerate and use)
   - partial reuse provisions
   - diagnostic procedures (what to do if a

problem occurs)
- usage examples

6. PROCUREMENT AND SUPPORT
- source (if not in library)
- ownership (any legal or contractual restrictions)
- maintenance (what support is available; points of contact)

Figure 1 InformationcontentoftheNATOre-user'smanual.(1993,p.8-5)

Normal documentation does not fully meet the needs of the RSCre-user; additionalsupportshouldbeprovidedinare-user'smanualforeverycomponent. The manual should follow a standard format.

B. The REBOOT Component Model (Karlsson, 1995)

According to Karlsson, are usable components apart of a product at some level of development (requirements, design, code), together with information about the component to make reuse feasible. There usable component must be self-contained. Hence, when a company has decided on which components to reuse, a decision on how these components should be packaged for reuse must follow. The component model describes the information needed for a packaged reusable component.

Karlsson provides an entity-relationship-based description of his component model with small components only parts of the model may be regarded as useful. The classification information aids component identification and retrieval.

The component qualification information describes the quality and reusability of the component. Information is used while deciding whether the candidate component fulfills requirements for quality and reusability. This information also tracks there use history of the component, i.e., the experiences and problems.

The component administrative information includes general information about the component, including authorization and prizing. Documentation comprises two kind s of information. First of all, it holds documentation to support the reuse of the component. Secondly, it holds information intended for the documentation of the production which the component will be included. Documentation supporting reuse

a) Enables the evaluation of each component,

b) Enables the understanding of the functionality of the component, and

c) Enables the adaptation of the component for specific needs. The component interface describes the boundaries of the component. The component body describes the internal workings of the component. The test support includes readily available test suites for the component.

The component model also defines relationships between different elements of the model. The realizes-relationship relates analysis, design and the resulting code of a component and this way reflects the possibility of component s existing on different levels of abstraction. The includes relationship relates one or more code components to form a composite object. The relationship shows components version history by linking different versions of a component.

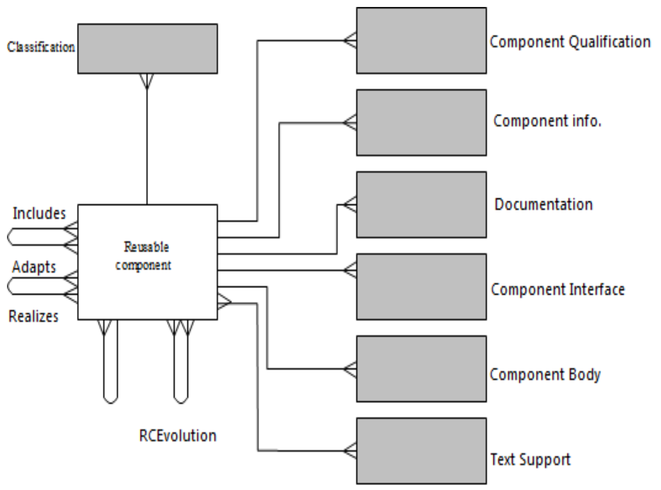

Figure: 2 The REBOOT component (documentation) model.(Karlsson,1995)

C. Sametinger's (1997) Reuse Documentation

In addition to the documentation of software there must be reuse documentation for software components. To effectively and correctly reuse a software component there should be in formation that enables

- The evaluation of component
- The understanding of the components functionality,
- The use of the component in a certain environment, and
- The adaptation of the component for specific needs.

Regular software documentation does not fulfill these needs. The component is not reusable without proper documentation. Thus, documentation must be valued as an essential part of a software component. Sametinger has elaborated his Reuse Manual on NATO's(1993) standard and Karlsson's book (Karlsson,1995). Additionally, he has used also Krueger(1992) and Meyer (1994) as the main references.

CONCLUSIONS

Short comings of the models

Each model basically recognizes the same general purpose for component documentation that supports the re use process. However, some differences can be found in how the models relate to the overall field of software documentation. The NATO (1992) model emphasizes that, in addition to explicit guidance to the potential re-user, component documentation must fulfill "the traditional role of documentation", Sametinger(1997) clearly distinguishes between other software documentation and his component reuse manual. In Karlsson's(1995) REBOOT model, there use-related aspects are partially embedded in the component documentation, and partially documented elsewhere.

Karlsson introduces separate documents for qualify in go administering components (which naturally support their use of the component as well).It seems that no clearly enacted demarcation yet exists for the genre system of component documentation.

All three models concentrate mainly on depicting the information content of component documentation. Sametinger's and the NATO model provide thorough guide lines about what information should be included, where as Karlsson's model stays on a more abstract level. Anyhow, differences exist, especially on how documentation is structured and what kind s of information are included under particular topics. Hence, a standardized structure for and a detailed set of document general to be included in this general system remain to be enacted. Moreover, these three models shed practically no light on how, by whom, and in what order the parts of component documentation should be created and updated. The sequence of the parts of component documentation thus needs to be better illustrated in order to increase general understanding of this (obviously rather complex) general system.

Little is said about what is expected from communication media and other issues of the actual form concerning this general system. The models seem to assume that component documentation mostly consist of digital text. NATO(1992) explicitly emphasizes that he documentation should be in a machine-readable form, in dependent of any particular word processor. Sametinger (1997) and NATO(1992) both highlight conformance with existing general-level documentation standards, and emphasize the fact that the writing should generally be clear, understandable, and complete. The lack of detailed responses to the how-aspect (especially communication media) seems natural because this aspect concretizes only when actual component repositories with adequate documentation are implemented and studied in real organizational contexts.

In all models, the primary stake holderis "there user" (NATO,1992,Karlsson,1995) or "the software engineer" (Sametinger,1997). In addition, Karlsson and the NATO model explicate, respectively, that "the library staff" and "the repository managers" also need to use component documentation. However, these models consider the question of who should produce the documentation either self-evident (i.e., the producer of the component is always assumed to produce the documentation as well) or other wise too trivial to be discussed. Karlsson, however, adds that the reuser should produce comments on previous use of acomponent to be included in the documentation, thus implying the possibility for various contributing take holders. The who-aspect of component documentation remains rather un-problematized. Furthermore, none of the models explicates temporal aspects of component documentation.

With regard the physical and logical location of the information, all three models denote that acomponent documentation should constitute a self-contained unit, i.e., it should not be embedded in one big document describe in gaset of components (Sametinger,1997) or other surrounding documentation (NATO,1992). The models also assume that component documentation should be placed in a logically organized (digital) repository. Interestingly, NATO(1992) points out that the actual (code) component can also be physically located else-where, for instance in a separate organization, as long as the documentation includes information about this location.

To summarize, the three models have focused mainly on the information content to be included in component documentation, neglecting the communicative view point to a large extent (which, however, represents the major rationale for the documentation in the first place).

REFERENCES

[1] T. J. Bigger staff and A. J. Perlis, editors "Software Reusability", vol. 1: Concepts and Models. Addison-Wesley, 1989.

[2] Zoran Stojanovic "An Integrated Component-Oriented Framework for Effective and Flexible Enterprise Distributed Systems Development Systems", Engineering Group Faculty of Technology, Policy and Management Delft University of Technology Jaffalaan, The Netherlands. (1999)

[3] P. Popov, L. Strigini, S. Riddle and A. Romanovsky, "Protective wrapping of COTS Components", NJIT University, USA.(1997)

[4] M. Woodman, O. Benediktsson, B. Lefever and F.Stallinger, "Issues of CBD Product Quality and
Process Quality", Annals of Software Engineering, 349-414. (1998)

[5] D. Garlan and B. Schmerl, "Component-Based Software Engineering in Pervasive Computing Environments", IEEE workshop, 403-444. (1996)

[6] SbengZhong, "Software Library for Reuse-Oriented Program Development", University of Windsor, Windsor, Ontario, Canada. (2000)

[7] I. Crnkovic, M. Chaudron and S. Larsson, "Component-Based Development Process and Component Lifecycle", In International Conference on Software Engineering Advances (ICSEA). (2006)

[8] J. M. Voas, "Certifying Off-the-Shelf Software Components", IEEE Computer, 353–59. (June 1998)