Analysis of lifecycle models and Software Component Retrieval
Monisha Kumari¹,Dimple Nagpal²

Abstract—The development of the software component consists of many stages and the lifecycle models show the flow of the stages that the development of the component follows. The development of the new software from the reusable software component has been challenging as it is always a concern to retrieve the relevant software component from the repository. There are lifecycles for making the design and development of the component more efficient. In this paper, an overview of the software component lifecycle models X-model, Y-model, Z-model, knot model and elite model and phases explanation of the lifecycle has been discussed. The current concept explains the comparison of the three lifecycle models in this paper. The component retrieval algorithms has been compared and discussed for the retrieval of the software component.

Keywords—Component Based Development (CBD), Component lifecycle models, Component retrieval, Component reusability, Software component.

I. INTRODUCTION
The component based development deals with the concept of reusing the already existing components. There are number of phases in the development of the software and the construction with the existing components has leads to many changes in the development phases.

The component based development follows various phases in the development.

1.1 Requirements Phase
In CBD the requirement phase consists of the analysis of the specification what the system expected to work. The goals, constraints and component analysis have been the part of requirement phase. This approach can be linked with the example in which obtaining the dining table by order from the carpenter or by purchasing it from the furniture shop. In second case one cannot get the exact result of dining table one wishes to have. In this way the software component gathering that can fit in the software development process is the major part of the requirement phase.

1.2 Analysis & Design Phase
The analysis phase will decide the component that can be fitted in the system development. This phase includes the detailed design of the system in which the relation between the requirements of the system are identifies and described.

In CBD the components are designed and developed using framework using the component models which is specified and decided in this phase.

1.3 Implementation Phase
In the implementation phase the design of the system formalize into the executable form. There are two important steps for this phase: 1) Selection of the component 2) Integration of the component with the system. In the first step the component is selected and verified. In second step the integration of components in assemblies tested because invisible dependency that can be resources shared or data shared can make the system fail.

1.4 Integration Phase
In this it checks how the components in the system are integrated and figure out the possibility of integrating of other components in the system for better results.

1.5 Test Phase
In this the software components in the system to be integrated are tested and results of the test are delivered to the software developer along with the component. The components to be tested are stored in the testable component repository.

1.6 Release Phase
The component are delivered and released along with the documentation, test results and test procedures. The components are delivered in the form of package that is suitable for the installation and distribution.

1.7 Maintenance Phase
The maintenance phase includes the repair of the component in the system if the component doing the malfunctioning then it can be replaced with the new component.

1.8 Coding and Archiving
In coding the code consists of steps that have been decided in the design phase written in any programming language which makes the steps understandable to the machine. In archiving the component storage has been involved and figures out the component to be well documented, understandable and well written.

¹Research scholar,Chandigarh Engineering college,Landran
²Faculty, CSE Deptt, Chandigarh Engineering College, Landran

Published online: 01/09/2016
1.9 Select component according software analysis

The selection of the component is done when the specification of the component is close to the requirements of the client. Afterwards selected component undergoes through adaptability and evaluation phase.

1.10 Evaluate & adaptability

The evaluation and adaptation is necessary phase in C_{BD}. The components are developed that tends to satisfy the needs of particular software that has been component based. So, to make the component adaptable the developer needs to do some changes in specifications, standards and environments.

1.11 Component wrapping and archiving

The power of the software developer to change the component that can be reused is known as component wrapping. There are three types of component wrapping (see table 1):

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>W_{BW}</td>
<td>I_P</td>
</tr>
<tr>
<td>G_{BW}</td>
<td>C_P</td>
</tr>
<tr>
<td>B_{BW}</td>
<td>P_P</td>
</tr>
</tbody>
</table>

W_{BW}=white box wrapping, G_{BW}=grey box wrapping, B_{BW}=black box wrapping, I_P= needs internal processing details of component, C_P=Application programming interface or component extension language provided by component library, P_P=component interface pre processing and post processing

1.12 Assembly

In this phase the components that are reusable are integrated into the architectural style. The components tend to be managed and interconnected in the appropriate infrastructure.

1.13 Domain engineering

In^2 Domain engineering is the process in which the application domain is analysed to identify the common areas and describing ways with help of uniform vocabulary. It is concerned with the identification of the components that are potential.

1.14 Domain analysis and specification

The components that are extracted from the domain are categorized and investigated in this phase. The classes that can be reused and the classes that are broadly applicable are analysed in this phase.

1.15 Frame working

In frame working the overview of the reusable components and their interrelationships within the application domain has been specified.

II. COMPONENT LIFECYCLE MODELS

The main approach of component based development is building a new application from already existing components. In^2 there are various steps for the development of the system with already existing components (see table 2).

<table>
<thead>
<tr>
<th>Steps</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step-1</td>
<td>Analysis of already developed component</td>
</tr>
<tr>
<td>Step-2</td>
<td>Finding and evaluation of the components</td>
</tr>
<tr>
<td>Step-3</td>
<td>Verify changes in component activities</td>
</tr>
</tbody>
</table>

2.1 VModel

V-model is very much disciplined model and beneficial for the small organizations with understandable requirements. The model is based on the waterfall model. V-model follows the bottom up approach for the component development and after completion of one stage the next stage starts. (See Figure 1). The V-model is suitable for the systems with following specifications: 1) Clear, well defined and documented requirements, 2) The project should be small. 3) The requirements should not be undefined or ambiguous. 4) Stable definition of product. 5) Project team easily understands the technology that should not be dynamic.

![Fig 1. V-model](image-url)

S_{C}=software analysis & specification, D_{E}=Design, C_{A}=coding & archiving, C_{T}=component testing, M_{A}=Maintenance, I_{D}=implementation deployment, S_{T}= system testing, A_{S}= assembly

INSIST Vol. 3 No. 2, October 2018 (149 - 153)
2.2 X Model

In the design of the X models is asymmetrical as the task is divided into constructive work and destructive work. The left hand side would be for constructive work and right hand side would be for destructive work. There are two phases in X-model. First is build of reusable components for development of application. Other one is introduction of build of application from testable as well as reusable component as shown in figure 2.

Testable component repository contains the component after 1) Testing 2) Development for reuse 3) The development with modification and without modification if applicable according to the lifecycle models

In X-model is based on four approaches 1) component development for reuse. 2) Component based software development 3) software development with reusable component modification 4) development of software without any reusable component modification.

2.3 Y Model

In Y lifecycle model mainly focus on the concept of reusability from its core for the development of the software. The components are developed that can be reused for the software development. The reusability is concerned with selecting the component and assembling it to make the software (see figure 3). This model is very much beneficial for the large software development and it takes ideas from both bottom-up and top-down approach. This model can be used for both development for reuse and development with reuse. In this model the previous knowledge of the software developer regarding the application domain is accounted. When the knowledge is limited the bottom-up approach is followed and otherwise most of the times top-down approach is followed. There are two abstraction layers in the Y-model. Component templates are consisted by upper layer and behaviour depicting run time object is consisted by lower layer.

2.4 Knot Model

In this knot lifecycle the reusability is the main focused concept. There are are four phases in the knot model. 1) Pool of reusable component 2) New component development 3) Existing component modification 4) New component based software development
mental processes that has been shown in Figure 4.

There are two funda
dvelopment of the software based on the reusability of the component. There are two fundamental processes that has been shown in Figure 4.

2.5 Elite Lifecycle Model (ELCM)

In the elite lifecycle model is based on the flow of the development of the software based on the reusability of the component. There are two fundamental processes that has been shown in Figure 4.

III. COMPARISONS OF LIFECYCLE MODELS

In this paper the comparisons of lifecycle model has been done as in the Table 3 and Table 4.

TABLE 3

<table>
<thead>
<tr>
<th>Phases</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pₚ</td>
<td>All the components stored in this including final product and other three phases have direct contact with it.</td>
</tr>
<tr>
<td>Nₖ</td>
<td>Development of component occurs when existing component in the resource pool does not satisfies the requirements of system, design and specification.</td>
</tr>
<tr>
<td>Eₖ</td>
<td>Modification of component occurs when existing component in the pool satisfies the does satisfies the requirement of system design and specification.</td>
</tr>
<tr>
<td>Nₐ</td>
<td>In this phase after risk analysis and testing all the component that are present in the repository are assembled</td>
</tr>
</tbody>
</table>

Pₚ= Pool of reusable component,Nₖ= New component development,Eₖ= Existing component modification,Nₐ= New component based software development

TABLE 4

<table>
<thead>
<tr>
<th>Model name</th>
<th>Shape</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>V₂ₕ</td>
<td>Vₘ</td>
<td>Eₘ, Hₘ</td>
<td>Aₘ, Hₘ</td>
</tr>
<tr>
<td>Y₂ₕ</td>
<td>Yₘ</td>
<td>Rₘ, Bₘ</td>
<td>Oₘ, Rₘ</td>
</tr>
<tr>
<td>X₂ₕ</td>
<td>Xₘ</td>
<td>Aₘ, Cₘ, Rₘ</td>
<td>Mₘ, Cₘ, Lₘ</td>
</tr>
<tr>
<td>Kₙ</td>
<td>Ω</td>
<td>Rₘ, Cₘ, Rₘ, Lₘ</td>
<td>Dₘ, Mₘ</td>
</tr>
<tr>
<td>Eₙ</td>
<td>Nₖ</td>
<td>Rₘ, Cₘ, Pₘ</td>
<td>Lₘ</td>
</tr>
</tbody>
</table>

V₂ₕ=V-model, Y₂ₕ=V-shape, Eₘ=Easy and simple to understand, Hₘ=highly disciplined, Aₘ=high risk and uncertainty, Bₘ=Asymmetrical design, Cₘ=Cost increase, Lₘ=Lack of risk analysis, Mₘ=More complexity, Dₘ=Difficulty in Selection of a right component, Pₘ=large reservoir and management is difficult, Rₘ=Reduces risk and development time,Nₖ=not applicable, Ω=Knot shape, Lₘ=applicable on larger & complex systems, Cₘ=Clear design, Nₘ=not feasible, Pₘ= perspective on election.

TABLE 4

<table>
<thead>
<tr>
<th>Development for Reuse</th>
<th>Nₖ</th>
<th>Nₖ</th>
<th>Nₖ</th>
<th>Nₖ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Component Based Software</td>
<td>Nₖ</td>
<td>Nₖ</td>
<td>Nₖ</td>
<td>Nₖ</td>
</tr>
<tr>
<td>Development without modification</td>
<td>Nₖ</td>
<td>Nₖ</td>
<td>Nₖ</td>
<td>Nₖ</td>
</tr>
<tr>
<td>Development with modification</td>
<td>Nₖ</td>
<td>Nₖ</td>
<td>Nₖ</td>
<td>Nₖ</td>
</tr>
</tbody>
</table>

Iₖ= Included, Nₖ= Not included

Fig 3. Y-model

Sₘ=Software analysis & specification, Dₘ=Design, Cₘ=coding & archiving, Cₙ=component testing, Mₙ=Maintenance, Iₙ=implementation deployment, Sₙ= system testing, Aₙ= assembly, Pₙ=frame working, Dₙ=domain engineering, Dₙ=domain analysis and specification.
IV. SOFTWARE COMPONENT RETRIEVAL

The software components have been stored in the repository and retrieval from the repository can be done by using different techniques of retrieval. In63 as the research done in context of component retrieval it has been developed that there are six retrieval algorithms (1) full text retrieval (2) keyword based (3) signature based (4) semantic based (5) Faceted based (6) enumerated based.

Other hybrid techniques for better retrieval of software component.

REFERENCES


IV. SOFTWARE COMPONENT RETRIEVAL

The software components have been stored in the repository and retrieval from the repository can be done by using different techniques of retrieval. In63 as the research done in context of component retrieval it has been developed that there are six retrieval algorithms (1) full text retrieval (2) keyword based (3) signature based (4) semantic based (5) Faceted based (6) enumerated based.

IV. SOFTWARE COMPONENT RETRIEVAL

The software components have been stored in the repository and retrieval from the repository can be done by using different techniques of retrieval. In63 as the research done in context of component retrieval it has been developed that there are six retrieval algorithms (1) full text retrieval (2) keyword based (3) signature based (4) semantic based (5) Faceted based (6) enumerated based.

IV. SOFTWARE COMPONENT RETRIEVAL

The software components have been stored in the repository and retrieval from the repository can be done by using different techniques of retrieval. In63 as the research done in context of component retrieval it has been developed that there are six retrieval algorithms (1) full text retrieval (2) keyword based (3) signature based (4) semantic based (5) Faceted based (6) enumerated based.

IV. SOFTWARE COMPONENT RETRIEVAL

The software components have been stored in the repository and retrieval from the repository can be done by using different techniques of retrieval. In63 as the research done in context of component retrieval it has been developed that there are six retrieval algorithms (1) full text retrieval (2) keyword based (3) signature based (4) semantic based (5) Faceted based (6) enumerated based.

IV. SOFTWARE COMPONENT RETRIEVAL

The software components have been stored in the repository and retrieval from the repository can be done by using different techniques of retrieval. In63 as the research done in context of component retrieval it has been developed that there are six retrieval algorithms (1) full text retrieval (2) keyword based (3) signature based (4) semantic based (5) Faceted based (6) enumerated based.

IV. SOFTWARE COMPONENT RETRIEVAL

The software components have been stored in the repository and retrieval from the repository can be done by using different techniques of retrieval. In63 as the research done in context of component retrieval it has been developed that there are six retrieval algorithms (1) full text retrieval (2) keyword based (3) signature based (4) semantic based (5) Faceted based (6) enumerated based.