International Journal of Advanced Trends in Computer Applications
www.ijatca.com

Review of models Analysis & Techniques of Software Component and Optimization with Genetic Algorithm

[1]Baljeet Kaur Nagra, [2]Jasneet Kaur
[1]Research Assistant, Computer Science and Engineering department,
Chandigarh University, Punjab India
[2]Assistant Professor, Computer Science and Engineering department,
Chandigarh University, Punjab India
[1]baljeete7255@cumail.in, [2]jasneete7747@cumail.in

Abstract: *This paper emphasizes on the highlights of the components reuse from Stone Age years to modern years and the comparative analysis of retrieval techniques. Methods/Statistical Analysis: There are models and techniques that are adapted in various researches for component retrieval. Findings: In the paper, an overview of the component models and various techniques for the selection of component has been discussed. To optimize the selection procedure of the component the genetic algorithm has been used. Applications/Improvements: The current concept explained in this paper explains the optimization selection process of the component by using the genetic algorithm.*

Keywords: Component Based Development, Software component, Component reusability, Retrieval techniques, Component models, Genetic Algorithm.

I. INTRODUCTION

The component based development is subset of Software Engineering which actually concentrates on the reusability concept .The keyword software component referred to a particular software unit which have already set the structural properties and functional properties that can be independently integrated into the system. These set of properties are called interfaces and accepted by provider of interface and software or other component that interacts with component (provider) .Component Based development involves the steps to design and develop the system with reusable integrated parts (components). The component can be addresses as software component, business component, distributed component and many more. The software's component is stored in repository and relevant components can be fetched or accessed from repositories for reusability concept shown in Figure 1.

First time the Author named Doug Mcllroy has given consideration of software component. In 1968, he proposed the concept of utility libraries. The retrieval of components is important idea for improving and enhancing the productivities of product quality in the software development.

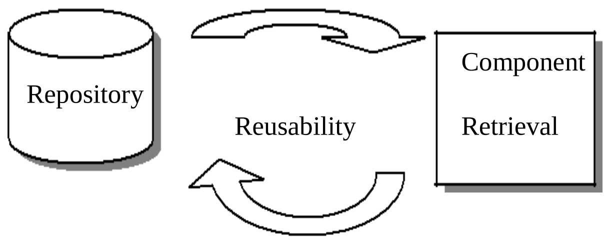

Figure 1: 0-Level DFD

II. EXISTING MODELS AND TECHNIQUES

The deployment components are technique which basically based upon the empowerment of the component integration into the existing system and the implemented deployed components are basically accessible to provider services. The process of binding the software deployed cycle which gives the interactive session between the components entrenched through interfaces and development system of the components is relied on the three principles: Reusability, Substitutability and Extensibility.

Reusability plays an vital role in the era which is extension to the device which can be reusing that reduces the build-up from basics. There are three main reengineering tasks in the retrieval process: 1) first is rely on partially specification mean how the component is beings searched and retrieved, 2) the reusability of components is worth of retrieving the component is accessed, 3) how to fit the reengineered components to the specific problem statements.

The component extensibility is supported by providing multiple interfaces as shown in Table 1.

Table 1: Three principles

Principles	Explanation	Feature
Reusability	Same component can be used again for different or same purpose.	Reduce build-up from basics.
Substitutability	Component accommodates with the system and meets requirements.	Maintain accuracy of the system
Extensibility	Component can be extended with the component which is part of the system or addition from outer environment.	Supported by providing multiple interfaces

1.1 TAXONOMIES OF COMPONENT

In table 1, the component can be business, technology type. These are explained in the Table 2.

Table 2: Taxonomies of component

Component Type	Instances
BCs	Auth, Iv, CRs, CCs
TCs	WSS, JCs, HCs.

BCs = Business based components, TCs = Technology based components, Auth = Authentication, Iv = Inventory, CRs=component repository, CCs = Credit card, WSS = web server based services, JCs = JSF based components, HCs = HSQLDB based components.

1.2 CONCEPT OF COMPONENT RETRIVAL

The retrieval concept for the component from the repository is mainly to find the relevant componential fundamentals from repository where large no components are present so that retrieved component can be reused. The main aim of component retrieval is based on the concept of "one time only one component from one repository". The extraction of the components from the repository is done by using the following technique as given below in Figure 2.

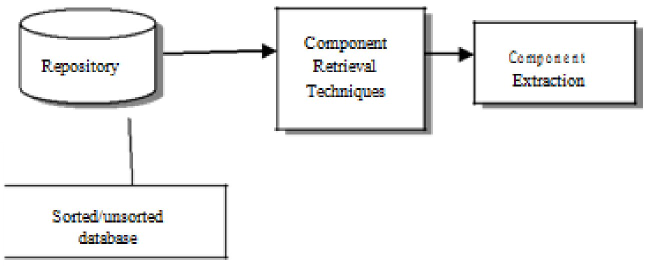

Figure 2: Level 1 DFD

1.3 COMPONENT'S EVOLUTIONARY STATES

The reuse of the components is a very vital process .in the traditional times the components like their weapons and tools were in the reusability process for different purpose of tasks to get the maximum optimization and today's techniques are based on those procedural fundamentals.

1.3.1 Components of Early year ages

In early Stone Age, The reusability process of tools, techniques and weapons as components has been initiated. Every tools and weapons had been used with the different ways and had the different work. The interaction among the people had been based upon the symbols which make their interaction keywords are based and the symbols were discusses in the families and relatives and those were basically the genetical algorithm based. The communications among the population were clustering and the association based.

1.3.2 Components of Modern Years ages

The idea of Re-engineering of software based component has been adopted from product based industry and structure fundamentals based presenting the design field. Production of motors and implementation of the fundamental of structures from all the blocks are called or known as cases.

The large industries already have taken or re-utilized a similar idea to create the algorithm or Sop. The industries has already provided their virtual products to market themselves effective which are basically the Programming parts and transported with the packages which are accessible with the particular software.

III. LITERATURE REVIEW

There are many approaches introduced that admit that the software components are better than object oriented ones for reusability. In[2] proposed the classification and retrieval of component by using genetic algorithm. In[3] describes the use of genetic algorithm for optimization. In[4] proposed the work on keyword based and genetic algorithm based technique. In[5] proposed the approach of component reusability by using the genetic algorithm. In[6] uses genetic algorithm and information gain approach for the purpose of the text categorization in the component principle analysis. In[7] proposed the project that uses the hybrid approach and evaluate the performance using precision and recall.

IV. EXISTING MODEL & TECHNIQUES

There are various models and techniques that have been developed by various research communities. These set of techniques provide various algorithm for better utilization of the components as in Figure 2.

COMPONENT RETRIVAL MODELS

In[8] describes that there are groups of models that had been developed by research communities and component analyst. The component model is a computer-aided system that defines the syntax, semantics and compositions of the component. There are various component models developed by researchers and analysts which includes EJB, .NET, CCM, WS, COM, KOL, CBR, SOF, ADL, UML, PEC, JB and FRC Table 3.

Table 3: Component Models

M_D	S_E	S_Y	M_D	S_E	S_Y
E_{JB}	C	OOPL	F_{RC}	O	IDL
J_B	C	OOPL	A_{DL}	AU	ADL
C_{OM}	O	IDL	U_{ML}	AU	ADL
N_{ET}	O	IDL	C_{BR}	AU	ADL
C_{CM}	O	IDL	S_{OF}	AU	ADL
W_S	O	IDL	P_{EC}	AU	ADL

MD = Models, SE = Semantic basics, SY = Syntax basics, C = Classes, O = Object, AU = Architecture Units, OOPL = Object Oriented Programming Language, IDL = Interactive Data Language, ADL = AD Language, EJB = EJB, JB = Enterprise JavaBeans, COM = COM,
.NET.= NET, CCM = CCM, WS = Web Services, KOL = Koala, CBR = CobrA, SOF = SOFA, ADL = Acme-like ADLs, UML = UML 2.0, PEC = PECOS, FRC = Fractal

The model recommends five phases to component reuse:
• Analysis of existing projects to sort components to be reused.
• Re-engineer to take out area particular inconveniences.
• Reusable components to be saved in the repository.
• Construct component with reusing approach and store in a repository with independent status.
• To develop new projects the components can be reused.

COMPONENT RETRIVAL TECHNIQUES

In[9, 10] proposed that there are different methods for seeking and the retrieval of the components for the efficient retrieval. The different techniques are shown in Figure 3 and explained:

Search method based upon free test

The TB is a process of searching the component in which the whole surrogate is looked and the relevant result discovered according to the search is shown.

Search method based don keyword

The KB is based on the method in which the client can just use keywords which as indicated by client are important as it would give the required component, once the keywords are entered in, the framework checks for just the keywords exhibit in the surrogate the required component, once the keywords are entered in, the framework checks for just the keywords exhibit in the surrogate and once the applicable keywords are found, the framework would show the relevant components.

Table 4: Search Methods

S. No.	Name	Techniques	Tools	Description	Pros	Cons
1	T_B	S_e I_n	B_{xa}, L_u	E_{xd}	I_{pr}	T_t
2	K_B	U_k	G_{kp}, K_t	P_c	F_r	L_{pr}
3	S_B	U_s	I_{ds}	S_f	H_{pr}	D_{ds}
4	O_{SB}	U_C	O_s	F_c	H_{pr}	D_{hd}

T_B = Free-Text Based Search, K_B = Keyword Based Search, S_B = Signature Based Search, O_{SB} = Operation Semantic Based Search, S_e = Searching, I_n = Indexing, B_{xa}

= BaseX, Algolio, L_u = Lucene, E_{xd} = Examines all the word in the document, I_{pr} = Improves precision and recall of the retrieval process, T_t = Takes lot of time, U_k = Use of keyword, G_k = Google keyword planner, K_t = keyword tool.io, P_c = One particular characteristic is used as a keyword, F_r = Faster results, L_{pr} = Lower precision and recall, U_s = Use of signature, I_d = Intrusion detection software, S_f = Some unique feature is considered as signature, H_{pr} = Higher precision and recall, D_{ds} = Difficult to decide the signature, U_c = Use of two-three characteristics, Os = Open Semantic Search, F_s = Two-three features are considered as operational semantic, H_{pr} = Highest precision and recall, D_{hd} = Difficult to handle non linear dependency.

Search method based on operational semantic

The OSB technique gives exact relevant component. In this search more than one attribute should be known by the client for this retrieval process. Subsequently exactness and review proportion is better for this situation as contrast with the above techniques.

In this case, the client will give as info that can be the components name and its model, and then the related component is shown with its info. The information of the input and output would be base points for this technique.

Search method based on signature

In SB, the search is depended on specific mark related to the component and the result would be approximately the exact match.

CLASSIFICATION SCHEMES

The classification based retrieval of component as described in[10] allows the client to retrieve or search more than one component from the relative same classified section. In[11] presents an overview about the principle inquire about on part seek and recovery in 2004 and examines how proficiently components can be inquiry keeping in mind the end goal to boost future component market. The main concentration of the work was grouping technique to store the product components.

To categorize the product components the feature has been used and the problem arises that two different people can describe the component with different keywords. In[12] give solution to this problem with the utilization of automatic indexing and extraction of terms or expressions that depict a best component. There are two fundamental characterization schemes: Faceted based and Enumerative scheme.

Faceted based scheme

The facet based scheme is explained as the classification technique based on the particular aspect of features. A particular facet value is assigned to the component at the time of component classification. The component can be identified by using the sets of facets like arrays, add, database manager, buffer as in Table 5.

Table 5: Classification schemes

Name	Applications
F_b	C_c, F_c, A_{at}
E_s	D_d, L_{cc}, U_d

F_b = Faceted based scheme, E_s = Enumerative Scheme, C_c = Clone classification for library materials, F_c = Faceted classification for occupational safety and health, A_{at} = Art and Architecture Thesaurus, D_d = Dewey Decimal system, L_{cc} = Library of Congress classification, U_d = Universal Decimal classification.

Enumerative Scheme

The enumerative scheme is discussed as a classification scheme which is based on the concept of mutually exclusive within progressive system of a solitary measurement

CLUSTERING TECHNIQUE

The clustering technique is one of the search and retrieval techniques which are based on the concept of gathering together the comparative components. There are two clustering techniques: 1) Self-Organizing maps (SOM) 2) Growing Hierarquical SOM (GHSOM). This technique refines the research results. When a developer searches for the java source code from the repository the developer will get the result along with the other similar component (source code).

BROWSING TECHNIQUE

One of the retrieval techniques is browsing, that has a requirement of well structured collection of document. The well structured document is represented as interconnected nodes network. The client can browse the required information through this network. The main advantage of this technique is when the user unable to decide the exact thing he is searching for. By using this technique, the user can go through the nodes consisting network which can help user for finding the exact match. This kind of matching doesn't have any formal rules.

ESISTING APPROACH FOR EFFECTIVE SEARCH

There are some existing approaches which makes the search effective

Knowledge Based Approach (KBA)
The KBA depends on the knowledge base which contains the semantic information of the component. The retrieval systems based on this technique are more rigid than the traditional system based upon the keyword approach.

Automatic Tag Extraction Based Approach (ATE)
This approach is based on technique to extract the component by using the tags. There are two steps in this approach: Firstly automatically the component tags are extracted and ATE algorithm is used to index it. Secondly similarity algorithm (vector space mode) is used to match the component tags in repository.

Automatic Indexing Based Approach (AIB)
In this approach the component is retrieved by using the index. The index contains the information of where the component is exactly present. There are two ways to do indexing either doing it manually or automatically by using computer.

GENETIC ALGORITHM (GA)

The Genetic algorithm is a technique that is discussed in this work for optimization that is based on population and various parameters.

The steps followed to retrieve the relevant component by applying Genetic algorithm is described below:

Component Repository
The component repository has been developed by the component developer. It contains the well defined components having some attributes and each attribute have some weight as in Table 6.

Table 6: Component Repository

Description	
Number of components in repository	N
Name of components	C_1 to C_n
Number of attributes	M
Number of weights	m

Calculate Fitness Function
Fitness function: The fittest individual in the population is determined by using fitness function. The population initialization is done by generating random value. The individual having higher value of fitness function would be most appropriate solution of the problem and the individual having the lowest value would be least appropriate solution. In the population every individual have its own value of fitness.

$= \sum^{n}{}_{=1} wt_i$………………… i.(1)

Fitness function of the component (C_{ff}) is calculated as the sum of weights (wt_i) of different attributes of the component.

Encoding
A binary string which represents the feature of the individual is known as chromosomes. The total feature number is equivalent to the length of the chromosome. The nth feature is stored in nth bit of the chromosome. If the value of the gene is "1' then the feature correspondence is selected otherwise the value of the gene would be "0". The fitness function optimization is done by encoded chromosomes after the initialization of population.

Table 7: Encoded values of C_i attributes

Attribute Values	Values
A_{v1}	001
A_{v2}	110
A_{v3}	101
A_{v4}	110
A_{v5}	011

Encoding of C_i = 001110101110011

The encoded values of attributes are extracted out for encoding the components into 0s and 1s (binary form). Suppose we have component C_i with 5 attribute values. The attribute values have some encoded values Table 7.

Selection
The individual selection for next generation is based on the two parameters: fitness function and selection method. Higher is the value of fitness function higher would be the chances of transfer of chromosomes to next generation. There are selection methods as Described in Table 8.

Table 8: Selection of components

Selection Method	Description
R_w	R_{wd}
T_s	T_{sd}
R_s	R_{sd}
E_s	E_{sd}

R_w = Roulettte-wheel Selection, T_s = Tournament Selection, R_s = Rank Selection, E_s = Elitist Selection, R_{wd} = Fitness-proportionate selection based on roulette game, each component gets slice of wheel and fittest will get larger slice., T_{sd} = From the large population subgroups consisting components are chosen and members of every subgroup compete each other and best from each subgroup are chosen to produce new population., R_{sd} = The selection is based on the ranking as each member in the population assigned a rank based on the fitness values., E_{sd} = Most fit member of a generation is selected.

Crossover

Crossover is a process to take two components chromosomes to produce a new component. Suppose there are two components: C_i and C_j. There are two types of crossover as in the Table 9.

Table 9: Crossover of components chromosomes

Component Name	Encoded Values
C_i	001110101110011
C_j	110001010011101
C_{ij}	0011101 10011101
C_{ji}	0011 0101001 1101

C_{ij} = Component after one- point crossover
C_{ji} = Component after two -point crossover

Mutation

Mutation is a condition applied on the chromosomes of the component to make it fittest as in the Table 10.

Table 10: Mutation of component chromosomes

Component name	Values
C_j	001101010011101
C_{su}	001111010011101
C_{in}	00111001010011101
C_{de}	1101010011101

Cji = component for mutation
Csub = component after substitution of 5th and 6th bit
Cins = component after insertion at 5th and 6th bit
Cdel = component after deletion of 1st and 2nd bit

TERMINATION

Termination (T_c) is a condition to stop the evolution of the population when convergence of fitness function (C_{vg}) occurs. The C_{vg} occurs when component retrieved satisfies the client or fitness value of component is equal to the best fitness value. But sometimes after number of iterations the t_c does not achieved. In this case generation number (G_n) is used to stop the evolution after maximum number of iterations.

The algorithm for the selection of component is described in Table 11 and optimization of selection result by GA is described in Table 12.

Table 11: Algorithm for selection of component

Algorithm: Component selection

```
• Algorithm for component Selection
• Input: Component Repository CR ; User
• Requirement UR
• Output: Component Select C_i
• Begin
• find_ true ;
• While CR_{} and find = true do
• Select UR_i € UR
• for i = 0 to n
• for j = 0 to m
• if (CR_j = UR_i )
• select CR_j
• break
• else
• Call Modify_Requirement(UR , CR , Result )
• return (UR' )
• endif
• endif
• call Create_Component(UR )
• end while
```

Table 12: Optimization

Algorithm: Optimization of selection result by GA

```
• begin
• t=0; initial time at the
• start of the algorithms
• Initialize population CR(t) component
• Repository
• Evaluate population CR(t) compute fitness of all
• initial component in the population
• While CR_ {} & find =true do
• t:= t+1
•  select CR(t) from CR(t-1) select sub-population
•  Crossover CR(t)
•  Mutate CR(t)
•  Evaluate CR(t)
•  End while
•  End;
```

V. CONCLUSION

In this paper the evolution of software component has been including, description about models and techniques of component retrieval has been described. There are effective retrieval techniques that can improve the performance by integrating with any one of soft computing techniques including Genetic Algorithm, machine learning, neural network, and Artificial Intelligence. In this work for making the retrieval more optimized the GA has been used.

References

W. Zhongjie, X. Xu, and D. Zhan. A survey of business component identification methods and related techniques. In IJIT, International Journal of Information Technology , 2006 Oct; 2(4),p.229

Andreou, S. Andreas, Vogiatzis D.G., and Papadopoulos G.A.. Intelligent classification and retrieval of software components. In 30th Annual International Computer Software and Applications Conference (COMPSAC'06), IEEE, 2006; p. 37-40, DOI:10.1109/COMPSAC.2006.135

Banerjee, Tania, Mohamed Gadou, and Sanjay Ranka. A genetic algorithm based approach for multi-objective hardware/software co-optimization. Sustainable Computing: Informatics and Systems 2016 June, (10); p. 36-47. DOI: 10.1016/j.suscom.2016.04.001

Bhatia, R. Kumar, M. Dave, and Ramesh C. Joshi. Retrieval of Most Relevant Reusable Component Using Genetic Algorithms. In SERP, Software Engineering Research and Practice, 2006 Jan; pp. 151-155.

Dixit, Anurag, and P. C. Saxena. "Software component retrieval using genetic algorithms." In Computer and Automation Engineering, 2009. ICCAE'09. International Conference on IEEE, 2009 Mar, pp. 151-155.

Uğuz, Harun. A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm. In Knowledge-Based Systems,2011 Oct, 7(24); p.1024-1032.

S. Vodithala and S. Pabboju. A dynamic approach for retrieval of software components using genetic algorithm. In 6th IEEE International Conference on Software Engineering and Services, 2015 Set; p. 406-410, DOI: 10.1109/ICSESS.2015.7339085.

Lau, Z. Wang and Kung-Kiu,. A taxonomy of software component models. On Software Engineering and Advanced Applications in 31st EUROMICRO Conference, 2005 Sept; p. 88-95.

Bakshi, Amandeep, and Seema Bawa. A Survey For Effective Search And Retrieval Of Components From Software Repositories. In IJERT International Journal of Engineering Research and Technology, 2013 April; 2(4).

Rao, CV Guru, and P. Niranjan. An integrated classification scheme for efficient retrieval of components. In JCS Journal of Computer Science, 2008; 4(10): DOI : 10.3844/jcssp.2008.821.825.

Daniel Lucredio, A survey on Software Components Search and Retrieval. In 30th EUROMICRO conference, 2004; p. 152-159.

YS Maarek , Kaiser GE and Berry DM, An Information Retrieval approach for automatically constructing software libraries, Software engineering based IEEE transactions , 1991 Aug; 17(8); DOI: 10.1109/32.83915.

William T, George T.,Councill and Heineman. Component-based software engineering. Putting the pieces together, addison-westley ,2001.

Patrick Hall, Barroca, Leonor, and Jon Hall. An introduction and history of software architectures, components, and reuse. In Software Architectures, Springer London, 2000; 10.1007/978-1-4471-0367-7_1.

Hamid /Mcheick., E. Ah-Ki, R. Godin, and Mili, Hafedh An experiment in software component retrieval. In IST Information and Software Technology, 2003 July;45(10) p.633-649

Kung-Kiu, and Z. Wang. Software component models. Software engineering based IEEE Transactions 10(33) , p.709-724.

Frakes, Thomas P. Pole and William B., An empirical study of representation methods for reusable software components. Software Engineering based IEEE Transactions, 1994 Aug; 20(8), p.617-630.

B. Ibrahim, Girardi, and M. R., An approach to improve the effectiveness of software retrieval. In the 3rd Irvine Software Symposium Proceedings, Irvine, California, 1993; 458.

Gen, Mitsuo, and Runwei Cheng. Genetic algorithms and engineering optimization. 2000; vol. 7.

C. Szyperski, Crnkovic, Ivica and J. Stafford. Software components beyond programming: From routines to services. In IEEE software, 2011 May/June; 28(03); DOI: 10.1109/MS.2011.62.