International Conference on Computer and Automation Engineering

Software Component Retrieval Using Genetic Algorithms

Anurag Dixit and P.C. Saxena
School of Computer and Systems Science
Jawaharlal Nehru University, New Delhi, INDIA

Abstract:

Maintaining consistency between the architecture of a component based software system and the components available in the component repository is a persistent concern in component-based software engineering . Nowadays in the software development practice, the percentage of time and effort allocated to this task is still too small to avoid bugs, which are late to discover and hard to fix. Various methodologies exist to annotate component selection models with data related to verification and validation, and to translate the annotated models into component performance. In any component selection method, it is unrealistic to expect a perfect match between components needed and components available. A group of components that compose a system may have overlaps and gaps in required functionality. A gap represents a lack of functionality; an overlap can cause a confusion of responsibility and degrade nonfunctional properties like size and performance. In the series of these technical unsolved issues, this proposed paper would be an attempt to throw light which on the one of the major issue of component based software engineering is concerned with the "Component Selection ".Recently a Genetic Algorithms based approach is used for component selection to Minimize the gap between components needed and components available. A relevant objective is now at hand in this direction, that is to make these methodologies acceptable from the software engineering community. Therefore, in this paper we develop Genetic Algorithms based approach for selection component.

Keywords: Component Repository, Genetic Algorithms**,** Fuzzy Data

1. Introduction

Component selection decisions are often made in an ad-hoc manner. Component selection processes have been proposed to improve upon the efficiency and effectiveness of informal methods. Existing selection processes do not fully address the specification and evaluation of functional and non-functional requirements[1]. Effective selection of reusable component in component based software systems is a hot issue for research community .A number of integration risks can often be resolved by selecting the 'right' set of components. One of the most critical processes in Component Based Software Engineering is the selection of the components from a reusable repository that meet the user requirements[2]. However, we have found that the information required evaluating those components using those quality models and metrics is not usually available in the existing commercial software repositories. In any component selection method, it is unrealistic to expect a perfect match between components needed and components available[3]. A group of components that compose a system may have overlaps and gaps in required functionality. A gap represents a lack of functionality; an overlap can cause a confusion of responsibility and degrade nonfunctional properties like size and performance. Software component retrieval is an area of active research in the field of component-based software engineering (CBSE). This paper presents a method that uses a semantic syntactic approach to retrieve components from a heterogeneous software repository called the software supermarket. A number of components are required to have a reasonable chance of finding matching components for a given task. Finding that component among all the others has been identified a major problem in component-based software engineering. This paper proposes a Genetic Algorithms based approach to satisfy such a need to its ability to naturally represent data selection. To support the development of high quality component-based systems, component selection processes need to address the problem of Functional and Non - Functional requirement evaluation of

978-0-7695-3569-2/09 $25.00 © 2009 IEEE
DOI 10.1109/ICCAE.2009.62

component-based software systems. This research proposes a component selection process, which is based on the Genetic Algorithms. Identification of motivation behind this research, Section 3 briefly describe the key element of component selection,

2. Motivation

Mismatch in component required and component retrieve from component repository is usually costly and possibly life threatening. Faulty component retrieval policy is costly in term of money , public safety and overall quality of system. As component based systems become more complex and increasing used in commercial as well as academics area , the cost of system failure continue to escalate . component retrieval is an important area of research aimed to producing more reliable systems. In the last decade, a lot of research work was performed in the field of component based software engineering for quality assurance and testing using Genetic Algorithms. Researchers showed that component based software methodologies can greatly improve by enhancing flexibility and mainability of systems. This approach can potentially be used to reduced software development costs , assembled systems rapidly ,and reduce the spiraling maintenance burden associated with the support and upgrade of large software systems. While this is certainly true , Integrator are still required to choose right component from component repository. This can leads to multiple , possible incompatible versions of component selection. Recently, researchers began analyzing inspection component repository to obtain insights on how component based software can be improved. Software Integrator needs to identify potentially error-prone/ require functionality component so that optimum quality system can be generated. Selecting the appropriate COTS component to meet specified Functional and Non - Functional requirements is a requirements assessment problem. It is widely agreed that requirements must be defined and quantifiable in order for selection of component to be effective. For the assessment of COTS component functional and candidate component is a complex activity itself. Section 2 briefly explain

Section 4 introduced the Genetic Algorithms and . A short discussion concludes the paper in Section 5. Non functional requirements must be specified and delegated to particular components in the software design. Existing component selection processes specify methods to elicit software requirements in the general sense, but they do not explicitly address how to specify requirements. Once requirements are specified, available COTS components must be evaluated to determine their suitability for use in the system being developed.

Finding the Right Component –

How are we to meet the software integrator's needs for finding, use and reuse components?

We argue that the matching process between component available and users requirement is a fundamental issue to be treated as a highest priority. In this context we propose a Genetic Algorithms approach to identify suitable component from component repository[5]. A research oriented Genetic Algorithms should have two basic properties.

-it should be easily customizable and extensible

- it should support software engineering principal

Any algorithms for component selection from repository are based on

(a)Best-fit Strategy: The Algorithms process is aimed to identify the best candidate component from the entire available component.

(b)First-fit Strategy: The Algorithms process is aimed to identify the first component that satisfies all of the user requirement.

Component Retrieval Steps:

```
Step1. Requirement Analysis
Step 2. Check  Specification of Component
 Step 3. Compare Requirement  with Component
Specification
Step 4 If match occurs select component otherwise
        elsif  modified requirement
      else  Construct new component
```

3. **Key Element:**

Component based system technology facilitates flexibility; systems can be adopted, modified and updated by adding, removing and changing their components. Adding or modified components require testing the components added or modified as well as all components that depend from them. Modifying components can introduce dangerous side effects that are difficult to predict, thus in principal, we need to retest the whole system. We define four key elements that must be considered during component selection .
(i)Specification (ii) Environment (iii) Interface
(iv) Events

Specification can be defined as the detail of component characteristic and its functionality.
Environment can be defined as the platform where component can work.

Interface can be defined as the access points of components through which a client component can request services declared in the interface and provided by another component. A component interface consists of two parts: A signature definition and behavior specification.

Event: Event can be defined as an incident in which an interface is involved in response to the incident.

4. Genetic Algorithms

A Genetic Algorithms is a search procedure modeled on the mechanics of natural selection rather than a simulated reasoning process. The approach is inspired by Darwin's theory of evolution, which is based on the survival of the fittest. A solution to a problem is considered an individual in a population of solutions. Domain knowledge regarding the problem is embedded in the abstract representation of a candidate solution termed as organism. Populations are sets of these organisms successive populations are called generations. Generations are produced by first identify the mating partners, and then applying the genetic operators on them to produced their offspring. Each individual so produced is checked for feasibility using a fitness function.

The methodologies have significant impact on the retrieval performance of a component based software systems. A variable selection that satisfies the coordination constraint , is the feasible solution to the design problem. These method for generating feasible component selection employ expensive computation and perform blind search to identify candidate component in the reuse component repository . A classification schemes based on a Genetic Algorithms were proposed to promote the qualities [7]. Effective reuse of component requires well –define repository. without these , the component repository becomes a write only storage medium. The repository of component is the link between Integrator, developer and user of the systems. The genetic approach views a variable organization as an organism with several layers of chromosomes. The populations of these component repositories are produced using genetic operators and each individual in these generations is checked for feasibility using a fitness function, which evaluates each individual component against user-defined criteria and structural and coordination constraints. Information obtained during this evaluation is used in genetic operators to direct a parallel heuristic search of the solution space for more fit component in the next (following) generations. The initial search space (population) of a Genetic Algorithms usually consists of solutions (individuals) generated randomly. With the initial population as the starting point, a Genetic Algorithms mounts a parallel heuristic search of the entire solution space for better solutions. The structure of a component Repository may be seen as a collection of independent several component capable of performing appropriate tasks. The inputs to the Component Repository (organization) are data from several sources of information, called requirement (sensors). This requirement (sensory) information determines which function is to be performed by the component .A component repository with each constitute chromosome representing user-defined constraints , shows the schema for the generation of future populations of structures. A 1 or 0 at any position means that the chromosomes in future populations must have the same value at that position for them to belong to the schema . The x's represent the genes (interactions) that can be replaced by either 1s or 0s genetically to generate new populations of solutions. The first step in the genetic algorithm approach requires an initial population of organisms to start the process. The bit strings representing the individual chromosomes are used to initialize the population. In order to feasible component repository , or a fit organism in genetic terminology , each Once we have all chromosomes in an organism represent feasible fixed structure , the organism is checked for coordination constraint.

4.1Genetic Algorithms

1. Initiate a population of chromosomes.
2. Evaluate each chromosome in the population.
3. Create new chromosome by mating current chromosome
4. Delete members of the population to make room for the new chromosome.
5. Evaluate the new chromosome and insert them into the population.
6. If time is up or get require , stop and return the best chromosome.

In the following section we present a novel approach for selecting candidate component from component repository.

4.2 Algorithm for component Selection

Input: Component Repository C_R ; User Requirement U_R

Output: Component Select C_i

```
Begin
find← true ;
While $C_R \neq \{\}$ and find =true do
 Select $U_{Ri}$ € $U_R$
for i=0 to n
   for j=o to m
if ($C_{Rj}$ = $U_{Ri}$ )
 select $C_{Rj}$
find ← false ;
break
else
 Call Modify_Requirement($U_R$ , $C_R$ , Result )
return ($U_{R'}$ )
endif
endif
 call Create_Component($U_R$ )
end while
```

4.3 GA for Component Selection

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

4.4 Binary –Coded GA:

Consider a Component Selection problem using binary-coded GA

In the binary-coded GA, the solutions are represented in the form of binary strings composed of 1s and 0s. A binary string can be compared to a biological chromosome.

Step1- Generation of a population of solutions:

An initial population of solution of size N

N may be 10 ,15,20,……..

Note:

1. N depending on the complexity of the problem

2. The length of a binary string based on a desired accuracy in the result.

Let N=4 its to small for GA Application but we only consider how to find best one

Chromosome label	Initial Population	Decoded Value
A	100100	36
B	011011	27
C	101010	42
D	111010	58

Step2: Fitness Evaluation:

Note: If the fitness is equal to the no. of ones in the strings ;Pselect= fi/∑fi

Initial Population	Value	Fitness fi	Pselect	Expected Count
100100	36	2	0.143	0.572
011011	27	4	0.286	1.144
101010	42	3	0.214	0.856
111011	58	5	0.357	1.428

A common selection method in GAs is fitness-proportionate selection, in which the number of times an individual is expected to reproduced is equal to its fitness divided by the average of fitnesses in the populations**.**

Step 3 : Reproduction:

All the GA- Strings may not be equally good in terms of their fitness value calculated in Step2. In this step , an operator named reproduction is used to select
In this step , an operator named reproduction is used to select the good ones from the population of strings based on their fitness information. There are several reproduction schemes have been developed by various investigators.

1. Proportionate Selection/Roulette –Wheel Selection
2. Ranking Selection
3. Tournament Selection
4. Elitism

We used Ranking Selection , there are only four binary string in Initial population of GA- solution ,where fitness value are 2,4,3,5 respectively. The process of ranking selection consists of two steps.
Step1: Initial Population are arranged by ascending order of their fitness value. The string having the lowest fitness is assigned rank 1 and other

Initial Population	Value	Fitness fi	Rank
100100	36	2	1
011011	27	4	3
101010	42	3	2
111011	58	5	4

Step 2: A proportionate selection scheme based on the assigned rank is adopted
Mutation

Step 4: Crossover:

In the crossover , there is an exchange of properties between two parents and as a result two children solutions are produced. There are number of techniques are present for crossover.

1. Single Point Crossover
2. Two Point Crossover
3. Multi Point Crossover
4. Uniform Crossover

Using single point crossover
100100 |011011
101010|111011

Two children produced due to the single –point crossover are:
100100|111011
101010|011011

Step 5 : Mutation :

In the GA , the mutation is performed by making a local change over the current solution . In mutation , 1 is converted into 0 and vice –versa.
The GA run until the termination criterion is reached that depends upon desired accuracy of solution.

5. Conclusions:

This paper suggests a new approach for component selection from component repository. Genetic Algorithms offer significant advantages over other approaches due to its ability to naturally represent qualitative aspect of inspection data and apply flexible inference rules. Our empirical studies show that retrieval of component is necessary yet expensive.

6.References:

[1] D'Souza D and A.C. Wills, Objects, Components, and Frameworks with UML: The Catalysis Approach, Addison Wesley Longman, Reading, Mass., 1998.
[2] Szyperski C, *Component Software: Beyond Object-Oriented Programming*, Addison Wesley Longman, Reading, Mass., 1998.
[3] G.T. Heineman and W.T. Councill, *Component-Based Software Engineering*, Addison-Wesley, Boston, 2001.
[4]Ioana Sora, Vladimir Cretu, Pierre Verbaeten, Yolande Berbers:Automating decisions in component composition based on propagation of requirements, in Fundamental Approaches to Software Engineering, Lecture Notes in Computer Science LNCS No. 2984, Springer, 2004, pp. 374-388
[5] J. Kontio: A case study in applying a systematic method for COTS selection.Proceedings 18*th* International, Conference on Software Engineering (ICSE), Berlin (Germany), March (1996).
[6] N. Maiden, C. Ncube: Acquiring requirements for COTS selection. IEEE Software 15(2), (1998).
[7] D. E. Goldberg ,Genetic algorithms in search, optimization & machine learning ,Addison Wesley, 1990.
[8] Chambers, L., , The Practical Handbook of Genetic Algorithms:Applications, Second Edition, Chapman & Hall / CRC2000.