

# Component system regression testing method based on CTAM

Xie Kai<sup>1</sup> Xu Baowen<sup>1,2</sup> Nie Changhai<sup>1,2</sup> Shi Liang<sup>1</sup> Zhang Xiaofang<sup>1</sup>

(<sup>1</sup> Department of Computer Science and Engineering, Southeast University, Nanjing 210096, China)

(<sup>2</sup> Jiangsu Institute of Software Quality, Nanjing 210096, China)

**Abstract:** The emphasis of component system regression testing is retesting of the event interaction between updated components and other components in a system. A component system regression testing method based on a new component testing association model (CTAM) is proposed. First, the modification-affected component groups are identified by the impact analysis on CTAM, and each component in this group is assigned with an influence degree. Then, previous test cases are selected according to the influence degree, to generate the minimal regression test suite. Compared with traditional methods, CTAM is derived from the statistic on the interactive events that occurred in previous test executions, and focuses on the complicated relationship between components, which is more applicable to the component system regression testing.

**Key words:** component; system testing; regression testing; test case selection

The component is an individual-functional, independently released binary module, which follows a certain component standard and can be integrated conveniently<sup>[1,2]</sup>. In the maintenance of the component system, requirements may change, which motivates the component modification to create new versions. When these updated components are integrated into the original system, large numbers of tests must be carried out. However, it will take too much cost and effort to rerun all the previous test cases. Meanwhile, because of components user's limited knowledge of third party component internal structure, it is hard to design effective test cases for a short time.

Based on previously run test cases, regression testing techniques can significantly reduce the retesting cost and effort. Traditional regression testing technologies mainly aim at structural programs and select test cases through code comparison<sup>[3,4]</sup> and program slicing<sup>[5]</sup>. However, these methods focus on the change of statements, unaware of the complicated interaction between components. We argue that, at a higher level, the component system regression testing is a retest based on previous integration and system test suites. So, the important key of the component system regression testing is not the mere retest for modified components' statements but the test to verify that new system fea-

tures have not destroyed existing components' interaction.

In the process of regression testing, the intensity and range of the test cases selection depend on the component association analysis. Our goal is to describe the component association by the component testing feature, and construct a new component testing association model (CTAM) on which we may be able to propose a complete component system regression testing strategy. The strategy includes modification identification, influence degree analysis and minimal regression testing suite generation.

## 1 Component Test Association Model

It is necessary for the component system regression testing to ascertain the association among requirement changes, modified components, modification-affected components and previous test suite. Here, we propose a formal CTAM, which is a weighted directed graph, with requirement node and component node respectively denoting components and requirements in the system; and weighted directed edges representing requirement-component association and component-component association between nodes. The model can reasonably characterize many of the component system architectures, and we will use it as a basis for the component system regression testing.

Let  $\text{Req} = \{f_1, f_2, \dots, f_n\}$  denote the requirement node in CTAM, where  $f_i$  denotes one of the functions in Req. The requirement node represents a set of discrete system requirements, which are divided by the specification. Each requirement is assumed to have the same granularity.

Let six-tuple  $\text{Com} = \langle \text{ID}, \text{CS}, \text{FS}, \text{PS}, \text{CTS}, \text{ES} \rangle$

Received 2005-04-20.

**Foundation items:** The National Natural Science Foundation of China (No. 60373066, 60403016, 60425206), the National Basic Research Program of China (973 Program) (No. 2002CB312000), Specialized Research Fund for the Doctoral Program of Higher Education (No. 20020286004), the Natural Science Foundation of Jiangsu Province (No. BK2005060).

**Biographies:** Xie Kai (1980—), male, graduate; Xu Baowen (corresponding author), male, doctor, professor, bwxu@seu.edu.cn.

denote the component node in CTAM, where ID denotes the component identification;  $CS = \{ID_1, ID_2, \dots, ID_n\}$  denotes the set of internal component identifications;  $FS = \{f_1, f_2, \dots, f_m\}$  denotes the set of component functions; PS denotes the set of component external interfaces; CTS denotes the component test suite;  $ES = \{e_1, e_2, \dots, e_l\}$  denotes the set of the test execution records, where triple-tuple  $e = \langle P_{tri}, P_{res}, n \rangle$  denotes the event which occurred in the previous execution of the component test,  $P_{tri}$  and  $P_{res}$  denote the interfaces of trigger and response component, and  $n$  denotes the times that the event occurred.

Fig. 1 shows the hierarchical structure of a general component system. In our model components of various layers are abstracted into component nodes. The component node also includes previous test cases for the component and test execution records.

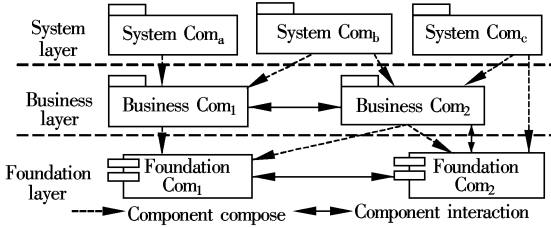


Fig. 1 Component system hierarchical structure

Let requirement-component association  $rRc$  denote the association between node  $Req$  and node  $Com$  in CTAM, iff  $Req \cap Com.FS \neq \emptyset$ , which is defined as

$$rRc(Req, Com) = \frac{|Req \cap Com.FS|}{|Req|}$$

Actually, a function defined in a requirement is realized by the collaboration of diverse components, while a component can output various functions defined in many requirements. We describe such association mapping the multi-function distribution in the component system, and assess it by the ratio of the functions output by a component to those defined in a requirement.

Let  $cRc$  denote the association between node  $Com'$  and node  $Com$  in CTAM, only if  $Com'.ID \in Com.CS$ . The component-component association  $cRc$  is defined as

$$cRc(Com, Com') = \frac{\sum_{e \in Com'.ES} e.n}{\sum_{e \in Com.ES} e.n}$$

There are many existing methods discussing the relationship between components such as interface definition language (IDL)<sup>[2]</sup>, abstract algebra<sup>[6]</sup>, unified modeling language (UML)<sup>[7]</sup> and so on. Nevertheless, these methods focus on structural dependent association from the aspect of component designing, unaware of the practical association between components in the previous test execution, which is critical to regression

testing.

We use event statistic, i. e. the ratio of events triggered or responded by an internal component to total events happening in the whole test execution, to assess the test association. Event mechanism is the basis of the component interaction, and there may be events of all kinds occurred in the execution of a test. Many commercial component systems implement a component transaction server (CTS) for the management of component event handling<sup>[2]</sup>. Fig. 2 illustrates that the events occurring in an executing test case can easily be recorded through the CTS interface.

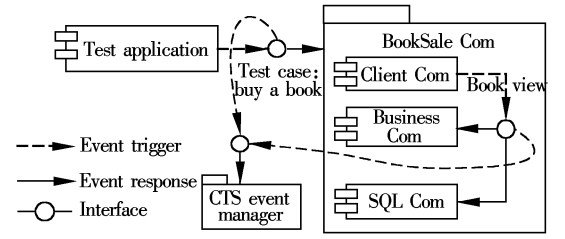


Fig. 2 Event occurring in component test

**Definition** Let quadruple-tuple  $CTAM = \langle NR, NC, L, w \rangle$  denote the CTAM, where  $NR = \{Req_1, Req_2, \dots, Req_n\}$  denotes the set of requirement nodes;  $NC = \{Com_1, Com_2, \dots, Com_m\}$  denotes the set of component nodes.  $L \subseteq NR \times NC \cup NC \times NC$  denotes the set of directed edges, where  $L(a, b) \in L$  denotes an existing directed edge from node  $a$  to node  $b$ ; and let  $w$  be the weight of the directed edge,  $L(a, b) \in L$ ,  $w(a, b)$  is defined as

$$w(a, b) = \begin{cases} rRc(a, b) & a \in NR, b \in NC \\ cRc(a, b) & a, b \in NC \end{cases}$$

The CTAM describes the function distribution in requirements and components and the hierarchical interactive structure of a component system. The directed edges from one node to another comply the order that begins at requirement nodes, relayed by component nodes of different layers from high to low, and ends at foundation component nodes. So it is apparent that the CTAM is an acyclic graph.

## 2 Modification Identification and Analysis

Selective regression testing is based on information about the changes made to the system<sup>[4]</sup>. So the identification and impact analysis for software change-affected parts are prerequisite conditions for the regression testing.

### 2.1 Modification-affected component group

The component system modification is mainly replacing the old component with a new one in the component library so as to adapt new requirements. For this reason, the most marked difference between component system regression testing and traditional regression testing is that the retesting objects are not the changed

program codes but the component interactions from among the modification-affected component group. So it is essential to determine such a group.

Requirement changes motivate the component's modification, which may consequently affect other interactive components. The interaction exists inside the component or between the components of layers. We divide the modification-affected components into three categories: ① A is the directly modified component influenced by requirement changes, and certainly A must be retested in the regression testing. ② B is the internal component of A, and thereby the modification to A will inevitably affect B. So B should be retested in the regression testing. ③ C is the component interacting with A. The modification of A causes the regression test for C, and then those components that interact with C

should be retested in the regression testing.

Thus, components satisfying the criteria compose the modification-affected component group (MACG) in the component system regression testing.

In CTAM the problem to determine the MACG caused by requirement changes is equivalent to finding the changed requirement node's reachable set in the graph. It is common that there is a series of requirements changed in the software maintenance. Let  $RC = \{rc_1, rc_2, \dots, rc_n\}$  denote the set of changed requirements, where  $rc_i$  is one of the changed requirements. Uniting all reachable sets of each  $rc$  in  $RC$ , we can derive the node set of MACG. In Fig. 3 the MACG affected by  $Req_a$  and  $Req_b$ , two changed requirements nodes in  $RC_1$ , is  $\{Com_1, Com_3, Com_4, Com_6, Com_7, Com_8, Com_9\}$ .

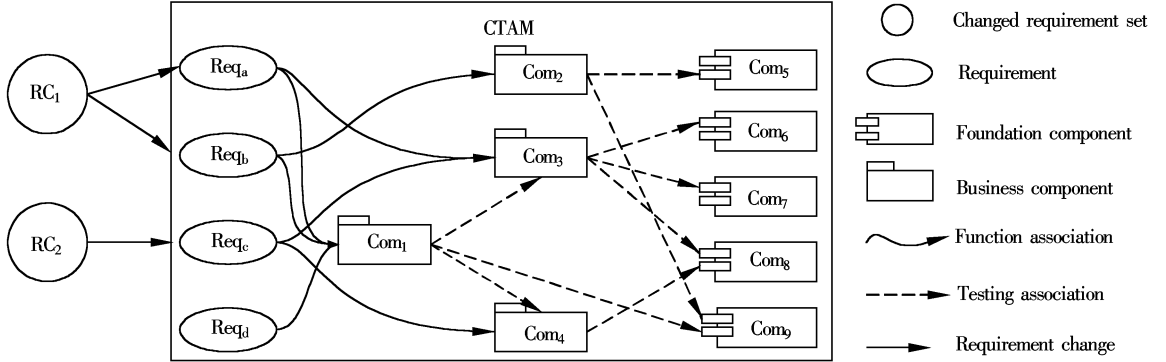


Fig. 3 Component testing association model

## 2.2 Modification influence measurement

MACG determines the range of components that must be retested in the component regression testing, but the influenced degree of each component in MACG is different. We present the modification impact analysis to assess the influence degree of each affected component.

Let node sequence  $P = (rc, Com_1, Com_2, \dots, Com_k)$ ,  $k \geq 1$ , denote the impact path from  $rc$  to  $Com_k$ , where  $rc \in RC$ ,  $Com_1, Com_2, \dots, Com_k \in NC$ . Let  $E(P)$  be the edge set of  $P$ ,  $E(P) \in L$ .

Originating from the changed requirements, the modification impacts are transferred from the directly modified components to other interactive components of different layers. We call such influence trace an impact path. In Fig. 3 two impact paths from node  $Req_a$  to node  $Com_8$  are Path 1:  $RC_1 \rightarrow Req_a \rightarrow Com_1 \rightarrow Com_4 \rightarrow Com_8$  and Path 2:  $RC_1 \rightarrow Req_a \rightarrow Com_3 \rightarrow Com_8$ .

Let  $PS = \{P_1, P_2, \dots, P_h\}$ ,  $h \geq 1$ , denote the impact path set from  $rc$  to  $Com_k$ , where  $P_i$  in  $PS$  denotes an impact path from  $rc$  to  $Com_k$ . Because CTAM is acyclic, the number of paths connecting two certain nodes is limited, and therefore the impact path set is a finite set.

Let  $Imp(rc, Com_k)$  be the impact degree that requirement  $rc$  affects component  $Com_k$ :

$$Imp(rc, Com_k) = \bigvee_{P_i \in PS} \left( \bigwedge_{L(u, v) \in E(P_i)} w(u, v) \right)$$

We use the connection degree in the weighted graph to measure the modification influence transferred along the impact path. The connection degree of the path is the minimum weight of all the edges of the path, and the connection degree between two nodes is the maximal connection degree of all the paths in the path set. The impact degree describes the component's dependence on the requirement. When the requirement is changed, such dependence decides how deeply the component is influenced.

Each changed requirement in  $RC$  has its own impact on the component  $Com_k$  in MACG. By summing all the impacts, we define  $Inf(RC, Com_k)$  as the modification influence degree of  $Com_k$ :

$$Inf(RC, Com_k) = \sum_{rc \in RC} Imp(rc, Com_k) = \sum_{rc \in RC} \left( \bigvee_{P_i \in PS} \left( \bigwedge_{L(u, v) \in E(P_i)} w(u, v) \right) \right)$$

The modification influence degree indicates the extent that those components supporting multi-functions are prone to be affected by requirement changes. Further-

more, if it is a component with high degree of modification influence, that should be enhanced by more selected test cases in the component system regression testing strategy.

### 3 Component System Regression Testing

Regression testing is an expensive and frequently executed maintenance process used to revalidate modified software. To improve it, regression test selection (RTS) techniques strive to lower costs without overly reducing effectiveness by carefully selecting a subset of the test suite. In this section, we propose a method for component system regression testing selection and a realizing algorithm.

#### 3.1 Regression testing selection

The simplest regression testing strategy is to re-run all existing test cases, which is simple to implement, but can be unnecessarily expensive. For a component system regression test, there may be a large numbers of components in the MACG, and it is impossible to reuse all the test cases that exercise these components. Therefore, a subset must be selected even further from the previous test suite.

Our approach is to assign the priority on each previous test case to increase or decrease the likelihood that it will be selected in the current regression testing suite. Two key factors judging this priority are the degree of the component's modification influence and the test case's component coverage in previously run test history.

The first factor means if a test case may exercise those components with high modification influence degree, the test case can be thought of as an "important" test case. We use the summation of the degree of modification influence on all the components covered by a test case, to measure its importance. The second factor makes attempts on selecting those test cases that cover as many as possible new components in the MACG. As an example in Tab. 1, we simply select Case<sub>1</sub> and Case<sub>2</sub> to cover the components Com<sub>2</sub>, Com<sub>3</sub>, Com<sub>4</sub>.

**Tab. 1** Component coverage of each test case

	Com <sub>1</sub>	Com <sub>2</sub>	Com <sub>3</sub>	Com <sub>4</sub>	Com <sub>5</sub>	...	Com <sub>k</sub>
Case <sub>1</sub>	✓		✓		✓	...	
Case <sub>2</sub>		✓		✓		...	
Case <sub>3</sub>		✓				...	✓
⋮							
Case <sub>n</sub>		✓		✓		...	✓

Based on the two factors above, we use heuristics to select constantly the most important and widest-covered test cases to generate the minimal component regression test suite. Let  $T$  denote the original test suite, where  $t$  is one test case in  $T$ , and let RTS denote

the selected regression test suite. At a high level, the test case selection works as follows:

**Step 1** Select all the test cases of  $T$  that cover the components in the MACG, yielding the RTS.

**Step 2** Draw a test case  $t$  from the RTS, so that the sum of the degrees of modification influence of the components that are both covered by  $t$  and marked "uncovered" in MACG, is the highest.

**Step 3** Mark the components in step 2 "covered" by  $t$ .

**Step 4** Repeat step 2 and step 3, until all the components in MACG are covered at least once.

#### 3.2 Algorithm

Based on the discussion above, we present the component system regression testing algorithm (CSRTA). The algorithm is composed of three phases: Phase 1 determination of MACG, Phase 2 impact analysis for MACG, and Phase 3 minimal regression test suite generation.

**Algorithm** CSRTA (CTAM, RC, TS): RTS

Input CTAM: component test association model

RC: changed requirements set

Output RTS: component regression test suite

```

begin
  RTS ← ∅
  MACG ← ∅
  for each rc ∈ RC                                     //Phase 1
    MACG ← DepthTraversal(rc, CTAM)
  for each com ∈ MACG do                               //Phase 2
    for each rc ∈ RC do
      PS ← ImpactPathGeneration(rc, com)
      for each P ∈ PS do
        pathimpact ← min{w(u, v) | L(u, v) ∈ E(P)}
        impact ← max{pathimpact(P) | P ∈ PS}
      endfor
      influence ← sum(impact from rc)
    endfor
  endfor
  for each com ∈ MACG do                               //Phase 3
    add com. TS to RTS
    mark com "uncovered"
  endfor
  repeat
    for each t ∈ RTS do
      for each com ∈ MACG covered by t
        priority ← sum("uncovered" com's influence)
      endfor
      select t with the highest priority
      for each com covered by t mark "covered"
    until all com ∈ MACG are marked "covered"
  return RTS
end

```

The analysis of algorithm CSRTA: ① The determination of the MACG in Phase 1 is a depth traversal in the CTAM, and the complexity is  $O(n)$ , where  $n$  is the number of the nodes in the CTAM graph. ② The main cost in Phase 2 is finding the impact path set between requirement nodes and component nodes in the

CTAM. Actually, it is equal to the problem of a possible path set between two nodes in a graph, and so far the optimum algorithm is Narahari Pandit's improved matrix algorithm and Floyd-Warshall's algorithm<sup>[8]</sup>. The former complexity is  $O(\alpha(n^3))$ , and the latter is  $O(2n^3)$ . ③ The component regression test suite generation in Phase 3 imposes on Harrold's heuristics method<sup>[9]</sup>, and its complexity is  $O(lm + \min(l, m)mk)$ , where  $m = |TS|$  denotes the size of the previous test suite, and  $l = |MACG|$  is the size of the MACG.

4 Conclusion

Traditional regression testing technologies are mainly based on the statement level. However, we argue that the component regression testing must be carried out at a system level, putting emphasis on the integration and interaction between the modified and modification-affected components. In this paper, we present a complete component system regression testing strategy based on a new CTAM. The testing association in the CTAM is derived from the test history record statistics, which can vigorously support component interaction analysis in regression testing. Components modification may affect a wide range of interactive components. So the size of the regression test suite selected by RTS techniques may consequently be extremely large. To reduce it, we propose the degree of computationally modification influence of each component, as a way of determining whether or not a test case should be selected in the regression test suite. Given that it may necessarily involve considerable resources to perform the impact analysis and minimal regression test suite generation, our further empirical research is needed to focus on the cost and effec-

tiveness prediction of component system regression testing.

References

[1] Weyuker E J. Testing component-based software: a cautionary tale [J]. *IEEE Software*, 1998, **15**(5): 54 – 59.  
[2] Edwards S H, Shakir G, Sitaraman M, et al. A framework for detecting interface violation in component-based software [A]. In: *Proceedings of the Fifth International Conference on Software Reuse* [C]. Victoria, Canada: IEEE Computer Society Press, 1998. 46 – 55.  
[3] Li Y J, Wahl Nancy J. An overview of regression testing [J]. *ACM SIGSOFT Software Engineering Notes*, 1999, **24**(1): 69 – 73.  
[4] Rothermel G, Harrold M J. A safe, efficient regression test selection technique [J]. *ACM Transactions on Software Engineering and Methodology*, 1997, **6**(2): 173 – 210.  
[5] Binkley D. The application of program slicing to regression testing [J]. *Information and Software Technology*, 1998, **40**(11, 12): 583 – 594.  
[6] Medvidovic N, Taylor R N. A classification and comparison framework for software architecture description languages [J]. *IEEE Transactions on Software Engineering*, 2000, **26**(1): 70 – 93.  
[7] Councill William T. Third-party testing and the quantity of software components [J]. *IEEE Software*, 1999, **16**(4): 55 – 57.  
[8] Yang L, Chen W K. An extension of the revised matrix algorithm [A]. In: *IEEE Int Symp Circuits and Systems* [C]. Portland, OR, 1989. 1034 – 1038.  
[9] Harrold M J, Gupta R, Soffa M L. A methodology for controlling the size of a test suite [J]. *ACM Transactions on Software Engineering and Methodology*, 1993, **2**(3): 270 – 285.

基于 CTAM 模型的组件系统回归测试方法

解 凯<sup>1</sup> 徐宝文<sup>1,2</sup> 聂长海<sup>1,2</sup> 史 亮<sup>1</sup> 章晓芳<sup>1</sup>

(<sup>1</sup> 东南大学计算机科学与工程系, 南京 210096)

(<sup>2</sup> 江苏省软件质量研究所, 南京 210096)

摘要: 提出组件系统的回归测试重点是对更新组件与其他组件之间以事件为单位的交互重新测试. 给出了一种基于新的组件测试关联模型(CTAM)的组件系统回归测试方法. 首先对需要进行回归测试的组件群体进行波动分析, 得出群体中组件受系统改动的影响度, 然后根据影响度选择复用先前的测试用例, 生成最小回归测试用例集. 与传统方法相比, 该模型建立在组件间交互事件统计信息的基础上, 分析了组件间的复杂关系, 更适用于组件系统的回归测试.

关键词: 组件; 系统测试; 回归测试; 测试用例选择

中图分类号: TP311