A Synchronized Test Control Execution Model of Distributed Systems

Salma Azzouzi, IBN TOFAIL University, Kenitra, Morocco

Sara Hsaini, IBN TOFAIL University, Kenitra, Morocco

My El Hassan Charaf, IBN Tofail University, Kenitra, Morocco

ABSTRACT

Conformance testing may be seen as mean to execute an IUT (implementation under test) by carrying out test cases in order to observe whether the behavior of the IUT is conforming to its specifications. However, the development of distributed testing frameworks is more complex and the implementation of the parallel testing components (PTCs) should take into consideration the mechanisms and functions required to support interaction during PTC communication. In this article, the authors present another way to control the test execution of PTCs by introducing synchronization messages into the local test sequences. Then, they suggest an agent-based simulation to implement synchronized local test sequences and resolve the problem of control and synchronization.

KEYWORDS

Controllability, Coordination, Distributed Testing, Observability, Synchronization

INTRODUCTION

In the conformance distributed testing context, the difficulty arises in controlling the execution of the test process. In practice, testing a distributed system, in order to ensure its conformance to the specification, involves usually placing a set of parallel testers called PTCs (parallel test components) attached to each port of the implementation under test (IUT). The difficulty is in guaranteeing the coordination between such PTCs. Hence, the design process should take into consideration the mechanisms and functions needed to support interaction, communication and coordination between the distributed components.

Many problems influencing faults detection during the conformance testing process arise if there is no coordination between distributed testers. In the main, two major problems known as controllability and observability fault detections occur while testing distributed systems. These problems have a great influence on several aspects of the testing activity, such as the execution of test sequences, the fault detectability of test system and the interpretation of testing results (Rafiq & Cacciari, 2003; Hierons, 2001; Ural & Whittier, 2003; Rafiq, Cacciari & Benattou, 1999; Tai, & Young, 1992). As solution to these problems, many works propose that testers exchange some coordination messages through reliable communication channels (Rafiq & Cacciari, 2003). However, many time-outs problems arise during the test execution due to the implementation of these communication channels which influences significantly the fault detection, this potential issue is called Synchronization problem.

DOI: 10.4018/IJGHPC.2020010101

The main based ideas of the proposed work are to develop an algorithm for generating Synchronized Local Test Sequences for each tester guarantying to avoid problems of coordination, observation and synchronization. Hence, the authors demonstrate trough their suggested proposition that the synchronization messages embedded in the local test sequences solve both problems of coordination and synchronization. Secondly, the authors have noticed -when they have implemented the SLTS during the simulation phase- a problem related to the order of reception of synchronization messages. To resolve this problem arisen during the testing process, they enhance their algorithm by adding some instructions to verify the conformance testing without taking into account the order when receiving the successive synchronization messages.

Finally, on high level of abstraction, the authors show how multi-agent-based system used in distributed testing prototype realization contributes to capture the complex monitoring tasks of distributed tester behaviors. Moreover, for the need of simulation, they realized also a distributed application based on the CORBA architecture (Common Object Request Broker Architecture) as an implementation under test which allows them to generate the outputs in response to a specific input as indicated in the global test sequence used as example.

The paper is structured as follows. Section 2 describes the architecture and the modeling concepts of distributed testing application. Section 3 presents some previous works proposed in the literature to overcome the distributed testing issues. Section 4 gives an overview of the test control and presents the algorithm allowing the generation of Synchronized Local Test Sequences used to resolve synchronization and controllability problems in the distributed testing implementation. Section 5 introduces a synchronized agent-based simulation for testing open distributed systems and the last section gives some conclusions and identifies future works.

DISTRIBUTED TESTING

The principle testing is to apply input events to the IUT and compare the observed outputs with expected results. A set of input events and planned outputs is commonly called a test case and it is generated from the specification of the IUT. Conformance testing may be seen as mean to execute an IUT by carrying out test cases, in order to observe whether the behavior of the implementation is conforming to its specification.

Architecture

The basic idea of distributed testing architecture is to coordinate parallel testers called PTCs (Parallel Test Components) using a communication service in conjunction with the IUT.

Each tester interacts with the IUT through a port PCO and communicates with other testers through a multicast channel (Figure1). An IUT is the implementation of the distributed application to test. It can be considered as a “black-box”, its behavior is known only by interactions through its interfaces with the environment or other systems. Each tester sends some stimulus to the IUT via their attached interfaces called PCOs (Points of Control and Observations) and from which it observes the output IUT reactions.

Test Procedure

To approach the testing process in a formal way, the specification and the IUT must be modeled using the same concepts (Dssouli, Khoumsi, Elqortobi & Bentahar 2017). The specification of the behavior of a distributed system is described by an automaton with n-port (Gill, 1962) (FSM Finite State Machine) defining inputs and the results expected for each PCO.

Σ_k denotes the input alphabet of the port k (PCO number k) and Γ_k the output alphabet of the port k. Figure 2 gives an example of 3p-FSM with Q = {q0, q1, q2, q3, q4, q5}, q0 initial state, Σ_1 = {a1,a2,a3}, Σ_2 = {b1,b2}, Σ_3 = {c1,c2,c3}, and Γ_1 = {x1, x2,x3,x4,x5,x6}, Γ_2 = {y1,y2,y3,y4}, Γ_3 = {z1,z2,z3,z4,z5}.