Future Generation Computer Systems 70 (2017) 74–93

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Acceptance Test for Fault Detection in Component-based Cloud Computing and Systems

Mounya Smara[a,*], Makhlouf Aliouat[b], Al-Sakib Khan Pathan[c,d], Zibouda Aliouat[b]

[a] Department of Computer Science, Faculty of Science, University of Ferhat Abbas Setif-1-, Setif, 19000, Algeria
[b] Laboratory of Network and Distributed Systems, Computer Science Department, University of Ferhat Abbas Setif-1-, Setif, 19000, Algeria
[c] Department of Computer Science and Engineering, Southeast University, Dhaka, Bangladesh
[d] Faculty of Computer and Information Systems, Islamic University in Madinah, Madinah al-Munawwarah, Saudi Arabia

HIGHLIGHTS

- Fault Detection in Component-based Cloud Computing using the Acceptance Test.
- Detection of transient hardware faults, software faults, and response-time failures.
- Theoretical comparison between the proposed framework and the existent strategies.
- Application of the Acceptance Test framework on the case study: Fire Control System.
- The efficiency of the proposed strategy is proved using the model-checker.

ARTICLE INFO

Article history:
Received 6 November 2015
Received in revised form
12 April 2016
Accepted 23 June 2016
Available online 26 July 2016

Keywords:
Fault detection
Component-based Cloud Computing
Recovery blocks
Acceptance Test
BIP framework

ABSTRACT

Fault Detection is considered as one of the main challenges in large-scale dynamic environments and thus, for maintaining the reliability requirements of Cloud and Mobile Cloud systems. Most of the popular existing techniques for fault detection applied on the Cloud Computing environment in general, are based on system-monitoring despite the extreme difficulty of keeping track of all machines with their huge number in Cloud systems. In this paper, we propose a Fault Detection framework for the Component-based Cloud Computing by using *Recovery Blocks*' Acceptance Test. This framework aims to construct *Fail-Silent* Cloud modules which have the ability of *Self-Fault* detection. In this, the detection process of transient hardware faults, software faults, and response-time failures is performed locally on each computing machine in the Cloud system. Background of the research issue, our mechanism, thorough analysis, and appropriate case study are presented. The efficiency and practicality of the proposed framework are proved by Safety verification using the model-checker.

1. Introduction

Embedded Computing systems could be seen now almost everywhere in our daily life. They are found in household items, multimedia equipment, in mobile phones as well as in cars, smart munitions, satellites and so on. However, despite increasing hardware capabilities, these mobile devices will always be resource-constrained compared to fixed hardware. In order to mitigate the hardware limitations on mobile and wearable devices, Mobile Cloud Computing [1–6] allows users to use remote infrastructure in an on-demand fashion. The size of the mobile market in consumer and enterprise is poised to reach over 45 billion Dollars by 2016 [7].

Mobile Cloud Computing is defined as the availability of Cloud Computing services in a mobile ecosystem [8]. Cloud Computing [9,10] is a type of parallel and distributed computing system which consists of a collection of inter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resource(s) based on Service-level agreements (SLAs) established through negotiation between the service provider and the consumers [9,11].

A Cloud application [11] is composed of a number of Cloud modules. Each Cloud module has a virtual machine used to realize its function and each function is composed of a set of *Tasks*. It

* Corresponding author.
E-mail addresses: smara_mounya@yahoo.com (M. Smara), aliouat_m@yahoo.fr (M. Aliouat), sakib.pathan@gmail.com (A.-S.K. Pathan), aliouat_zi@yahoo.fr (Z. Aliouat).

http://dx.doi.org/10.1016/j.future.2016.06.030
0167-739X/© 2016 Elsevier B.V. All rights reserved.

is evident that Cloud Computing architecture, its layers and its composition of components and services need to be designed as web service components [12] based on well proven Component-based Software Engineering. The Component-based Software Engineering [13] is a reuse-based approach to define, implement and compose loosely coupled independent components into systems. Here, a component represents an entity that provides a specific functionality. The components are expected to be scalable, fault tolerant, manageable, and autonomous [14]. Several tools are available for modeling heterogeneous embedded systems founded on component-based models. One of them is BIP tool. BIP (Behavior, Interaction, Priority) [15–20] is a tool for behavior-based modeling of heterogeneous real-time components.

We could fairly state that applications developed on Mobile Cloud systems are often critical in terms of human lives. For instance, many such applications could be practically employed in healthcare, military, or disaster management scenarios. Therefore, these types of systems must be reliable; in the way that can offer correct or acceptable services even in the presence of faults. In other words, critical applications must satisfy the *Safety* property [21]. Reliability is one of the main characteristics of Mobile Cloud systems [22,5] but it is difficult to analyze due to its characteristics of massive-scale service sharing, wide-area network, heterogeneous software/hardware components and complicated interactions among them. The failures that can occur in a Mobile Cloud environment can be classified into two classes [23]: Data failures and Computation failures. Data failures are due to the exploitation of data whereas computation failures are due to hardware faults, software faults or network faults. The response-time failures can occur due to Data failures or Computation failures. In this paper, we deal with transient hardware faults, software faults, and response-time failures.

Fault tolerance techniques used in Cloud Computing [23–28] are based on time redundancy or spatial redundancy which can tolerate only hardware faults without dealing with software faults. According to our thorough investigation of the area, there is clearly a lack of formal approach that rigorously relates Component-based Cloud Computing with Software Fault Tolerance concerns.

Software fault tolerance mechanisms are based on design redundancy [29]. *Recovery blocks* technique is a type of *Forward Recovery*. It is based on the selection of a set of operations on which recovery operations are based. *Recovery blocks* is composed of a set of try blocks and an Acceptance Test. The Acceptance Test is a section of program which is executed on exit from the try block to confirm that it has performed acceptably [30,31]. A *checkpoint* is used for saving the last correct system state; it is used in the Recovery phase.

Our main objective is to incorporate Recovery Block's Fault Detection strategy in Component-based Cloud Computing in order to be able to develop Cloud modules which are *Self-Fault Detectors* using the *Acceptance Test*. In this paper, we propose a novel formal framework for constructing models apt for software, transient hardware and response time fault detection. Each Cloud module in the Cloud application must have an Acceptance Test which can validate its behavior. If the Cloud module behavior is correct or acceptable, it would continue operation otherwise, and it will be stopped immediately—then, we can say that we have a *Fail-Silent* Cloud module. Just to clarify here a bit, a *Fail-Silent* system is a type of system that either would provide the correct service, or would not provide any service at all (i.e., would become silent). We will use the semantics of BIP (Behavior, Interaction, Priority) [15–20] as a Component-based framework with multi-party interactions.

A framework for fault detection in Component-based models was first introduced in [32], but that was not used for Cloud system. Here, unlike the previous work:

- A background of the existent strategies for fault detection in Cloud systems with their strength and limitation is presented.
- The Acceptance Test framework is proposed for detection of Cloud and Mobile Cloud faults such as: Transient hardware faults, software faults, and response-time failures.
- Theoretical comparison between the proposed framework and the existent strategies is presented.
- The proposed framework is then applied on a Mobile Cloud case study: Fire Control System.
- Time and space complexity of the Acceptance Test strategy are estimated.
- A Safety verification by the model-checker is applied on the deduced Fail-Silent Fire Control model to prove the efficiency of the proposed strategy.

We have organized the rest of the paper as follows: following the introduction, an overview of the existing strategies for failure detection in Cloud Computing systems with their strength and limitation is presented. Then, our proposition for fault detection by using the Acceptance Test is introduced. First, some basic concepts of BIP are described. After that, we describe our approach for fault detection in Component-based Cloud modules. In this section, we prove that we can ensure *Safety* property in *Fail-Silent* components using the Acceptance Test. Then, we present how we can construct Fail-Silent components from basic BIP components (Atomic component, Composite component). Finally, we apply our approach on Fire Control System as a case study. Time and space complexity are calculated. Then, a Safety verification using the model-checker is applied on the deduced Fail-Silent Fire Control model to prove the efficiency of the proposed strategy. For all the sections, appropriate analysis and discussions are presented to highlight the effectiveness of our approach. Before concluding the paper with future research directions, we present all the major relevant works.

2. Failure detection in Cloud Computing systems

Failures in Cloud Computing systems are processed by using two main strategies: Intrusion detection and Heartbeat/Pinging.

2.1. Intrusion and anomaly detection systems for cloud

Intrusion Detection Systems (IDSs) [33–37] are strongly adopted in Clouds. Generally, IDSs are used for detection of network or hosts attacks (e.g., Denial of Service, Buffer Overflow, Sniffer attacks). They are based on *behavior observation* of the component and an alarm is raised if an abnormal behavior is detected. They can be grouped into two detection principles, namely misuse-based (or Signature-based) and anomaly-based IDS.

2.1.1. Signature-based IDS

This kind of IDS recognizes intrusions and anomalies by matching observed data with pre-defined descriptions of intrusive behavior. Therefore, a signature database corresponding to known attacks is specified a priori.

2.1.2. Anomaly-based IDS

The strategy of anomaly detection is based on the assumption that abnormal behavior is rare and different from normal behavior, and thus it tries to model what is normal rather than what is anomalous. Anomaly detectors generate an anomaly alarm whenever the deviation between a given observation at an instant and the normal behavior exceeds a predefined threshold (see Fig. 1). Anomaly Detection refers to the important problem of finding non-conforming patterns or behaviors in live traffic data. These non-conforming patterns are often known as anomalies.