IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 6, 2020, accepted February 21, 2020, date of publication February 27, 2020, date of current version March 12, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2976718

AUTOCON-IoT: Automated and Scalable Online Conformance Testing for IoT Applications

JAEYOUNG HWANG[1], ABDULLAH AZIZ[1], NAKMYOUNG SUNG[2], ABBAS AHMAD[3], FRANCK LE GALL[4], AND JAESEUNG SONG[1], (Senior Member, IEEE)

[1]Department of Computer and Information Security, Sejong University, Seoul 05006, South Korea
[2]Korea Electronics Technology Institute, Gyeonggi-do 13509, South Korea
[3]Department of Computer Science, Université de Franche-Comté, 25000 Besancon, France
[4]Easy Global Market, 06560 Valbonne, France

Corresponding author: Jaeseung Song (jssong@sejong.ac.kr)

This work was supported by the Institute of Information and Communications Technology Planning and Evaluation (IITP) funded by the Korean Government (MSIT) under Grant 2019-0-00426 (Development of active kill-switch and biomarker-based defence system for life-threatening IoT medical devices).

ABSTRACT As the Internet of Things (IoT) is evolving rapidly in various verticals, such as smart cities and smart industries, the number of IoT applications is growing exponentially. Standardized conformance testing mechanisms are widely used to test software applications, including IoT applications. However, current conformance testing mechanisms require continuous human intervention and additional software to be embedded into a target device being tested, which is time consuming, costly, and requires additional memory on the target device. Therefore, there is a considerable need to enhance and optimize existing conformance testing so that it is adequate for automatically testing highly distributed constraint IoT applications. In this paper, we propose a novel mechanism for automated and scalable conformance testing for IoT applications by introducing a test triggering mechanism based on a standardized test interface. This triggering mechanism is based on several new logical components in a testing environment that exchange messages between a testing system and a target IoT device. This technique of automatic conformance testing can lessen the cost and human intercession to decrease the number of missteps and accelerate the IoT market by certifying IoT applications.

INDEX TERMS Conformance testing, Internet-of-Things (IoT), interoperability, oneM2M standards, testing automation.

I. INTRODUCTION

The Internet-of-Things (IoT) is acting as the next technological revolution influencing all application domains, including smart homes, smart cities, agriculture, automobiles, healthcare, industries, and transport [1], [2]. It is estimated that there will be 50 to 100 billion smart things and objects connected to the Internet by 2020 [3]–[6]. For the last two decades, interoperability has been the main hurdle of IoT development and adoption [7]–[10]. To successfully adopt the IoT, it is necessary to overcome its fragmentation. An acceptable standard is required to ensure the interoperability between IoT services and eliminate fragmentation by introducing horizontal applications [11], [12].

The associate editor coordinating the review of this manuscript and approving it for publication was Yang Liu.

Assuring the interoperability between IoT implementations referring to the same standards requires a well-known conformance and interoperability testing process [13]–[17]. In this regard, IoT conformance testing is becoming one of the most important aspects of IoT technologies [18], [19]. The primary goal of conformance testing is to ensure that an IoT device has properly implemented its referring standard specifications such as detailed values of protocol messages and their format based on standard specifications. In general, conformance testing consists of a test system, a system under test and a set of test cases, where the testing system executes the test cases against a system under test to assess its degree of standard compliance.

Such conformance testing approaches do not work well with constrained IoT devices, which do not have any user interface (UI) or enough memory space to store testing-related source code. To test such constrained IoT devices,

many manufacturers use their own software agent having UIs for stimulating the system under test to initiate a specific behaviour. However, such testing tools require human developers to be involved in conformance testing procedures. For example, to test a registration function, which requires a system under test to send a registration request to a testing system, a developer should instruct the system under test via a software testing agent that sends the corresponding stimulus to the system under test. Testing hundreds of IoT constrained devices using such conformance testing tools is a time-consuming job, which is not efficient for testing IoT devices.

To automate the testing process for constrained IoT applications, two requirements should be considered in a testing approach for IoT applications: (1) the system under test should be instrumented by a testing system via a standardized testing interface without human intervention, and (2) a set of testing application programming interface (APIs) on the testing system that stimulates the system under test to initiate a specific behaviour. These requirements allow the test system to control constrained systems under test with minimum testing functions. To exchange the necessary information for testing, there is a need to introduce a set of logical components and procedures in both the test system and system under test to enable automated testing. For example, if a TS needs to test a device registration capability with a constrained system under test, a corresponding trigger message should be sent to the constrained system under test as a stimulus. As this message is sent over a standardized interface between the system under test and the test system, the constrained system under test can perform a proper procedure. In this case, the system under test sends a registration message to the test system.

In this article, we describe *AUTOCON-IoT*, a newly developed testing approach to automatically test a large number of constrained IoT applications. AUTOCON-IoT introduces a logical component called *UpperTester* that sits between the testing system and the system under test and automatically performs testing procedures. UpperTester uses a triggering mechanism that sends a standardized test message to a target system under test so that the system under test can initiate a specified function to be tested. In addition, AUTOCON-IoT uses a set of common testing APIs that defines different triggering messages (e.g., discovery, registration, and measurement) to confirm the compliance of the system under test. With the adoption of the AUTOCON-IoT into the existing testing system, it is predicted that testing experts can reduce the testing error resulting from human intervention, and these standard-based automated testing approaches can also be applied in other IoT specifications and industry fields.

We empirically evaluate a prototype implementation of AUTOCON-IoT with an IoT global standards called oneM2M. We can define a set of testing APIs using the triggering mechanism, which can test constraint oneM2M IoT applications without human intervention. Our experiments also reveal that the use of a triggering mechanism in the IoT testing improves the time required for testing and shows advantages over existing conformance testing using stimuli.

The main contributions of our work are as follows:

1) A testing approach that uses a standard-based triggering message that instruments a constrained IoT application under test to perform specified tasks and checks functional compliance of the IoT application. Our approach fully automates testing procedures for constrained IoT applications; therefore, many IoT applications can be easily tested without human intervention.
2) Our experience developing the approach, together with an experimental evaluation of a triggering-based IoT testing method with several conventional conformance testing mechanism shows the advantage of the approach in terms of performance and testing time.
3) Standard inputs to an industry driven de-factor standards group, the oneM2M IoT standards body, where the proposed triggering mechanism is reflected in one of the normative specifications as a standard-based testing mechanism. The triggering IoT testing feature based on the proposed approach in oneM2M has already been included in designated oneM2M testing tools.

The remainder of this paper is organized as follows: Section II provides an overview and state-of-the-art for conformance testing to motivate automated IoT testing. Section III describes an architecture for an automated and scalable conformance testing mechanism for IoT applications. Section IV shows the experimental results based on a proof-of-concept AE conformance testing tool followed by related work in Section V. Finally, Section VI summarizes this article and proposes our future work regarding the IoT standard conformance testing tool.

II. BACKGROUND AND MOTIVATION

We start by giving some background about conformance testing and standard IoT platforms that we refer to, followed by the motivation for the need for a new testing approach for constrained IoT applications.

A. OVERVIEW OF CONFORMANCE TESTING

Conformance testing is a branch of testing where an implemented system is tested against its standard specification [20]–[22]. The main goal of the conformance testing is to improve the confidence of the implemented system to enhance the probability that it follows the same standards while communicating with other implementations. Conformance testing is functional testing [23], where an implementation of a standard is purely tested with reference to its specification.

Although there exist many different methods for performing conformance testing, the European Telecommunications Standards Institute (ETSI) defines a high-level methodology with essential components for standard-based conformance testing as shown in Fig. 1 [24], [25]. The methodology is

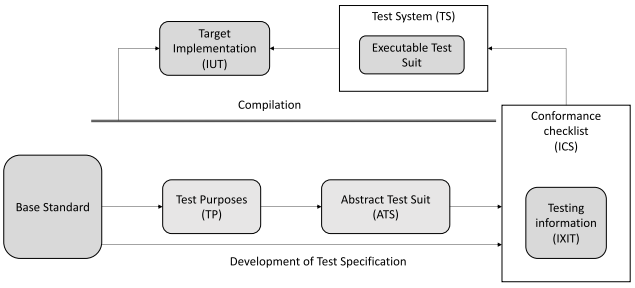

FIGURE 1. ETSI conformance testing.

composed of defining several specifications and developing relevant test cases as follows:

- An implementation conformance statement (ICS) is a checklist of the prospects defined in the standard. The ICS provides an overview of the features and options that are implemented by any given implementation under test (IUT). The ICS facilitates the tester in selecting and parameterizing test cases.
- The Implementation eXtra Information for Testing (IXIT) contains supplementary information, such as detailed addresses of test system (TS) and IUT and timer values, which are required for testing.
- Test purposes (TPs) [26] provide an informal, and an easily readable self-sufficient definition of each test, that focuses on what is to be tested instead of how a certain test may be accomplished.
- The abstract test suite (ATS) is a complete collection of test cases. Each test case should be written in a formal testing language to be executed. For example, the testing and test control notation (TTCN-3) [27]–[29] can be used as a language to which is the detailed coding of the TPs.
- An executable test suite (ETS) can be generated after compiling a TTCN-3 based ATS by using a TTCN-3 compiler [30], which is available on many test tool platforms.

In summary, based on functional requirements, a set of conformance testing specifications is needed to define the expected behaviour of a target system (e.g., IoT applications) and retrieve executable test cases. Then, a testing system uses the test cases to test the target system, for example, sending a request to the target and analysing whether a response contains expected information.

In the following sections, we refer to the conformance testing methodology in the oneM2M global standardization body. oneM2M IoT standard is widely used and implemented by different vendors. It is used throughout the paper to demonstrate automated IoT application testing by our approach. Additionally, our approach is integrated into the oneM2M testing specifications.

B. ONEM2M AND ITS CONFORMANCE TESTING

oneM2M [31] is a global standardization initiative to standardize a horizontal IoT service layer [32] that is network

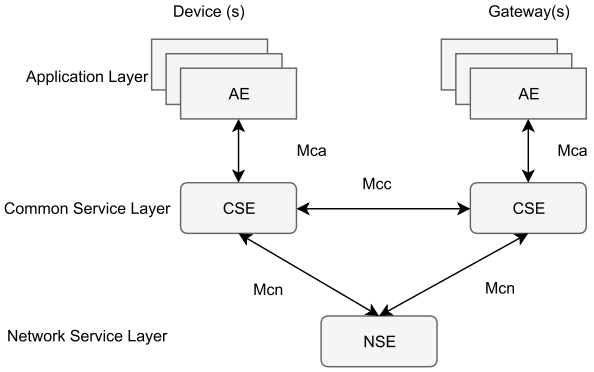

FIGURE 2. oneM2M layered architecture.

TABLE 1. oneM2M product profiles.

Profile	Description
ADN Profile 1	IoT application sensing data in a constrained IoT device such as a temperature sensor
ADN Profile 2	IoT application actuating things in a constrained device such as a dimmed light
ADN Profile 3	IoT application in a normal device such as a smartphone
ADN Profile 4	IoT application in a small originator device
IN Profile 1	Server device supporting IoT services
ASN Profile 1	IoT application in a normal actuator device
MN Profile 1	Gateway devices that support multiple different area network technologies and connect devices

independent and provides interworking to different current IoT vertical systems by reducing fragmentation in the IoT market. Fig. 2 shows the layered architecture of oneM2M. On the top of these layers, application entities (AEs) populate within distinctive devices, such as sensors, actuators, and the gateways. In the middle, common services entities (CSEs) perform an identical task in the service layer. In the application layer, the AE can be one of the four product profiles of the application dedicated nodes (ADNs) defined in oneM2M specifications, which include the application service node (ASN), the infrastructure AE (IN-AE), and the middle node AE (MN-AE). Similarly, in the common service layer, it can be the infrastructure node (IN-CSE) or the middle node (MN-CSE). The description of all these seven product profiles is provided in Table. 1. As these profiles are implemented by different manufacturers in various types of devices, a strong conformance testing program is needed.

C. ONEM2M CONFORMANCE TESTING

oneM2M standardizes a set of testing related specifications [33] to verify that a product is implemented based on the oneM2M specifications: the implementation conformance statement (ICS) [34], test suite structure and test purposes (TSS & TP) [35], and the abstract test suite (ATS) [36].

As shown in Fig. 3, oneM2M conformance testing consists of several steps as follows:

1) Gathering the test requirements and the product features to generate an implementation conformance

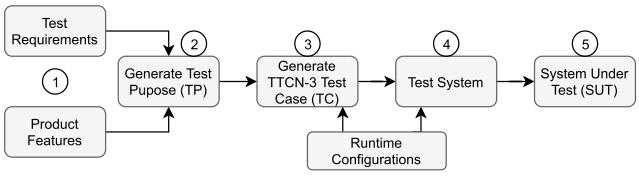

FIGURE 3. oneM2M conformance testing.

statement (ICS) that helps the tester to select and parameterize the test case.

2) Generating of TPs, and the group of all the TPs that will form the TSS.
3) When the TPs and the TSS are ready, the tester writes the test cases in a test description language such as TTCN-3.
4) The tester compiles the test cases using a TTCN-3 compiler and will configures the test system according to the runtime configurations.
5) Finally, the tester will make a connection between the test system and the system under test to perform testing.

D. CHALLENGES AND MOTIVATION

Fig. 4 shows how conventional conformance testing can be applied to constrained IoT applications that do not have a UI to control the behaviour of applications. Both the test system (TS) and system under test (SUT) start with normal steps, e.g., configuring the testing environment and preparing test execution. As a target SUT does not have any user interface, the manufacturer of the SUT typically brings its own software or tool that can control the SUT to perform a specific behaviour, e.g., sending a registration request. However, these tools cannot support interoperability, reusability, and integration with learning activities because each tool has its own data structure. Additionally, a tester of the TS has to inform the participants which test to perform manually. Manually performing these steps in a loop is a hectic and time-consuming job, and it leads to inevitable mistakes and errors in the test report [37]. Because a large number of IoT applications from different manufacturers or developers need such kinds of conformance testing, there is a strong need for standard-based automatic testing for IoT applications using stimuli.

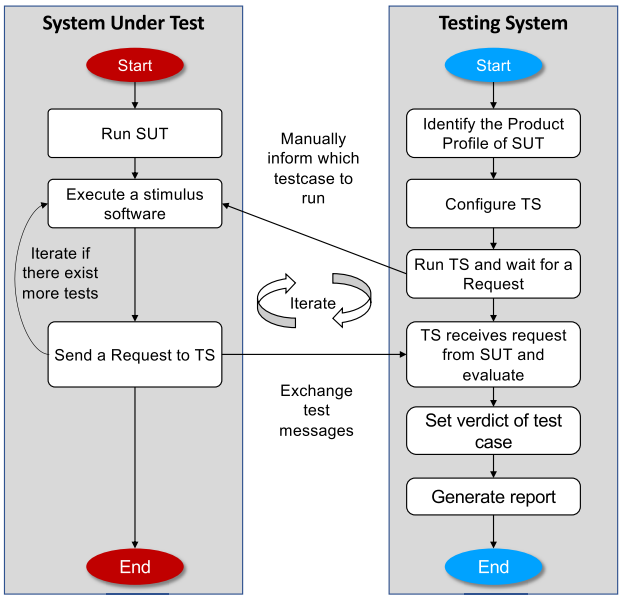

FIGURE 4. Flow chart of IoT conformance testing procedures between TS and constraint SUT that needs external stimulus software.

In the next section, we introduce an automatic conformance testing for IoT applications in constraint devices using a test triggering message, i.e., a test stimulus. The proposed mechanism is integrated into the oneM2M testing methodology so that it can be reusable.

III. STIMULUS-BASED AUTOMATIC CONFORMANCE TESTING

As mentioned in the previous section, we propose a standard-based automatic conformance testing for IoT applications using stimulus. The basic idea is to trigger an IoT application under testing to perform a requested behaviour so that a testing system can check its compliance. To make our approach applicable to IoT applications from different manufacturers, we design the testing framework based on oneM2M global IoT standards. In the next subsections, we present a system architecture of our testing framework as well as a triggering mechanism by introducing logical components in the SUT and the TS, a format of a trigger message, and a description of how it works.

A. THE ARCHITECTURE OVERVIEW

Figure 5 shows the architecture for automatic conformance testing of the IoT applications using the test stimulus. We designed AUTOCON-IoT based on the testing system standardized in the oneM2M conformance testing. Both the TS and SUT in AUTOCON-IoT contain the *upper tester* (UT) (ut-TS for the TS and ut-SUT for the SUT), which is an essential software component coordinating all the testing procedures, to support the automated conformance testing without human intervention.

AUTOCON-IoT starts its testing process with an input of a control file, *testConfig*, that provides all the necessary information to perform testing. testConfig contains a type of SUT, features implemented in the SUT, communication information to set up a test connection, testing protocol, a list of test cases to perform, and serialization formats. Under the assumption that the TS has required test cases written in a testing language such as TTCN-3 and codec to convert test cases into a supporting IoT protocol format, the TS executes each test case synchronously with the SUT. When a test case is completely executed, the TS executes the following test case from the control file based on a specific order. We define a standardized triggering message called *trigMsg* as a stimulus to be exchanged between the ut-TS and ut-SUT. For the TS to perform a particular test procedure, it has to deliver exact information to the SUT. For example, if it is time

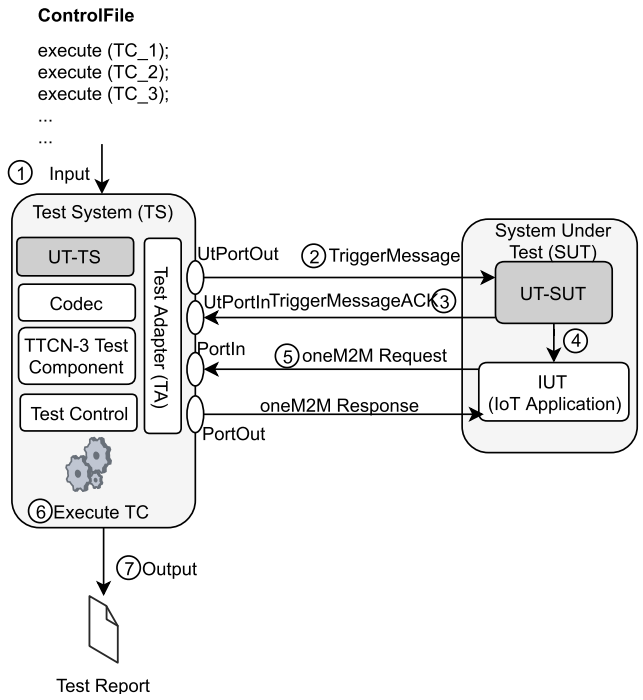

FIGURE 5. The architecture of automatic conformance testing.

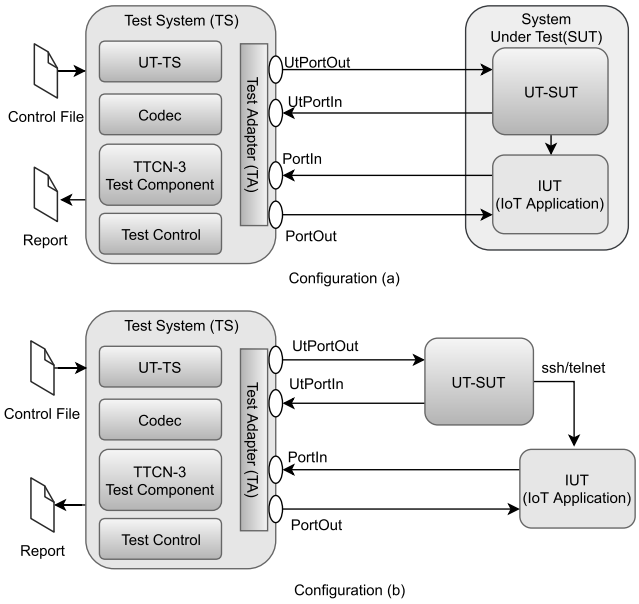

FIGURE 6. The configurations of automatic conformance testing.

Trigger Message Format

- operation: (mandatory) operation type that IUT is triggered to perform

- resourceType: (optional) resource type of a target resource against which IUT is triggered to perform a certain operation.

- to: (mandatory) address of the target resource against which IUT is triggered to perform a certain operation.

-primitiveContent: (optional) represents the resource attributes that shall be included in the requestPrimitive.

Resource Type

1 = accessControlPolicy
2 = Application Entity (AE)
3 = Container
4 = ContentInstance
5 = Common Service Entity (CSE)
23 = Subscription

Operation Type

1 = Create
2 = Get
3 = Update
4 = Delete
5 = Notify

FIGURE 7. The triggering message format.

for testing the registration function of the SUT, a registration request message should be sent to the TS from the SUT. The TS guides this information to the SUT using trigMsg via message exchange between the ut-TS and ut-SUT.

UT is the key testing component that forms the triggering mechanism and removes human intervention by automating the conformance testing process of IoT applications. Depending on where UT is located, it behaves differently as follows:

- *ut-TS*: The upper tester in the TS acts as a client and is responsible for generating a post request of the triggering message as indicated by the test case in the control file and sending a triggering message to the ut-SUT.
- *ut-SUT*: The upper tester in the SUT is a lightweight server that receives a triggering message from the ut-TS and coordinates the testing behaviour of the SUT. It parses a received triggering message to retrieve the data from the test case in the message body and initiates the predetermined operation.

The UT in the SUT is a lightweight testing server, which is appropriate for rich IoT applications that have enough computing power. However, not all IoT applications have such resources, and they are not equipped to deal with any additional software component such as the ut-SUT. In these resource-constrained IoT devices, the ut-SUT cannot reside inside the SUT. To address this particular issue of testing the IoT applications that run in resource-constrained devices, AUTOCON-IoT supports two configurations based on where to locate the ut-SUT. If an SUT has enough computing power and resources, as shown in Fig. 6 (a), ut-SUT is hosted in the SUT. However, for constraint IoT devices, ut-SUT can exist outside the SUT, which is depicted in Fig. 6 (b). For this situation, ut-SUT communicates with the TS in an indistinguishable manner. However, to communicate with the IoT application in the SUT, the ut-SUT needs to know the IP address of the target IoT application to establish a network connection for testing. Then, the ut-SUT can invoke a specific operation from the IoT application according to the trigger message from the TS. There are various approaches to setting up communication between ut-SUT and the IoT application, for example, SSH and Telnet.

B. TRIGGERING MESSAGE

The key to executing automatic conformance testing is to define a standard-based triggering mechanism between the TS and the SUT. The trigger mechanism in AUTOCON-IoT depends on two coherent components over the UTs in the TS and the SUT. ut-SUT acts as a lightweight HTTP server that runs in SUT, and it is comprised of one exposed

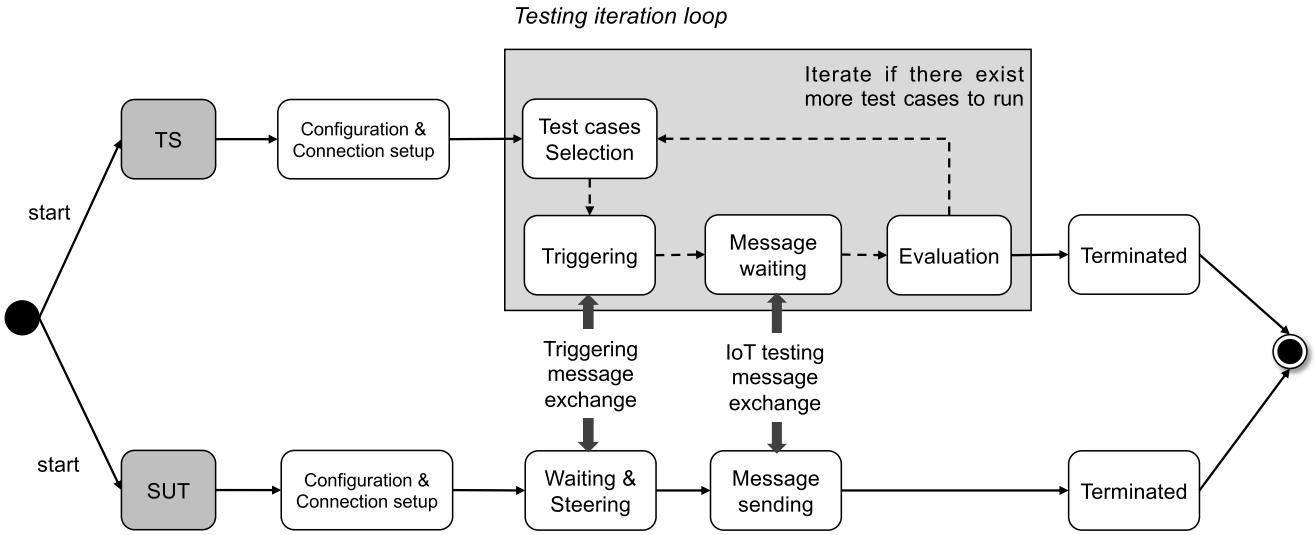

FIGURE 8. Finite state machine of AUTOCON-IoT showing automatic conformance testing for IoT application.

endpoint to a defined request-response message system expressed in a specific message format such as JSON or XML (https://www.w3.org/TR/REC-xml/). In this section, we consider JSON as a message format. This lightweight HTTP server is exposed to a local network where it cannot obtain incoming requests from external sources. A representational state transfer (REST) is the underlying architectural principle of ut-SUT, which exposes only one endpoint with one POST method. To make a legitimate request, the client needs to include four parameters: the uniform resource locator (URL), method, a list of headers, and the body of the request as a JSON object. The URL is an IP address of ut-SUT with only a single exposed endpoint as a root (/), which serves only as a post method of the HTTP request with standardized headers and body format according to referring IoT specifications such as oneM2M.

The body of the request contains the information identified with the test case and the TS in the JSON format which is shown in Fig. 7. Let us look at how a triggering message can be formulated in the oneM2M IoT standard. The body of the request has at most four key/value pairs, such as the operation and target as mandatory parameters, while a resource type and a primitive content are optional parameters. The operation parameter is numerically enumerated on numerous operations that are characterized in the oneM2M standard specification as 1=`Create`, 2=`Get`, 3=`Update`, 4=`Delete`, and 5=`Notify`. Accordingly, there are six resource types defined in the oneM2M specifications, and they are identified as:

- 1 = `accessControlPolicy`
- 2 = `Application Entity`
- 3 = `Container`
- 4 = `ContentInstance`
- 5 = `CommonServcieEntity`
- 23 = `Subscription`

The response format of the triggering message is composed of a response code. If the triggering message is correctly formatted by ut-TS, then the response from ut-SUT is `OK_REQUEST` (2000), otherwise it is `BAD_REQUEST` (4000).

C. AUTOMATIC TESTING PROCEDURE

The actual procedure of the automatic conformance testing is similar to the procedure shown in the flowchart in Fig. 4 with a few changes, such as the tester needs to configure the points of address and communication ports for both ut-TS and ut-SUT. Typically, when a product is presented for conformance testing, the test administrator needs to determine and list the test cases in the control file, as demonstrated by the chosen product profile features. In most cases, such control files are prepared statically, which is indicated by the features of a testing product profile.

As shown in Fig. 8, AUTOCON-IoT takes the control file as an input to TS and operates both the TS and the SUT in the same local network. At this time, ut-SUT starts listening for a triggering message from ut-TS. Based on the information in the control file, ut-TS chooses a test case and formulates a triggering message. The message is delivered to ut-SUT as an HTTP POST request. ut-SUT parses the request body by executing a message analysis algorithm and initiates the specific operation on the IoT application according to the parsed information. The analysis algorithm checks the validity of the request and then extracts the values of the mandatory attributes, which are the target resource in the TS and an operation type. Furthermore, if the optional attributes exist in the message body, it obtains the resource type information and the primitive contents.

If ut-SUT resides inside the SUT, it can directly execute the requested function in the request message. Otherwise, ut-SUT needs to establish a network connection with the target IoT application via lightweight communication means such as SSH and telnet. After establishing the network connection, a proper command needs to be sent to the IoT

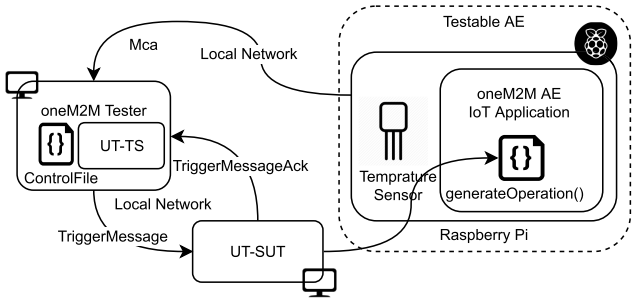

FIGURE 9. AUTOCON-IoT experiment configuration.

application to trigger the requested operation. Then, the IoT application sends the predetermined request to the TS to be assessed by running the test case and setting the verdict against the specific test case.

IV. EXPERIMENT

This section shows the experimental results of an automatic conformance testing process for the IoT applications. To show the feasibility of AUTOCON-IoT, we compare the proposed automated IoT testing with conventional conformance testing.

For testing purposes, a normal IoT application to be tested needs to support ut-SUT. We implemented a constrained IoT application with ut-SUT based on Profile 1 from Tabl 1 to examine the proposed mechanism. The test code implemented for the testable IoT application contains the `generateOperation()` function to receive a command to be executed from ut-SUT. The command is then executed in the IoT application according to the declared operation in the message. In addition, we implemented a TS system embedded with ut-TS using an IoT conformance testing system called *oneM2MTester* (https://github.com/IoTKETI/oneM2MTester), which was developed based on open source Eclipse Titan for oneM2M platform developers.

In the experiment, ut-SUT receives a triggering message from ut-TS and parses its data to obtain the values for intended operation, target, resource type and primitive contents. After validating the triggering message, ut-SUT responds with an acknowledgement message to the TS. ut-SUT also creates an SSH connection with the IoT application to communicate and invoke operations, such as the `generateOperation()` function. Finally, the `generateOperation()` function executes the testing procedure from its IoT application according to the received operation value from ut-SUT. TS then evaluates the test oracle after receiving the triggering request from the IoT application and determines a verdict.

In the case of test cases, we use standard-based IoT test cases developed by the oneM2M testing working group, as our triggering mechanism is reflected in oneM2M test cases. The oneM2M test cases are all written based on TTCN-3 and available publicly to be used for testing purposes.

The actual payload for the triggering message is shown in Lists. 1 and 2. This payload contains the parameter `op`: 1, which indicates a CREATE operation, `ty`: 2 indicates the IoT application resource type, the `to` parameter points to the target resource address in our TS, and `pc` is an object that contains primitive contents with respect to the oneM2M standard specification. The receiving IoT application uses this `pc` object while sending the post request to the TS.

```
{
 "m2m:rqp" :{
   "op": 1, //indicate CREATE operation
   "ty": 2, //indicate AE resource type
   "to": {TEST_SYSTEM_ADDRESS},
   "pc": {"m2m:ae": {
                        "lbl":"UNINITIALIZED"
   //indicate that attribute labels needs to be
       included
                        }
         }
     }
}
```

Listing. 1. Excerpt of triggerMessage format.

```
exports.create = function (URL,ty,body){
  AE.mandatoryAttributes(body, function (body){
  var headers= AE.headers
  request({
      headers: headers,
      url:TS_IP+URL,
      method:'POST',
      body:JSON.stringify(body)
    },function(error, response, body)
    {
      if(!error)
      {
        var obj = JSON.parse(body);
        console.log("Response=",obj)
      }
      else
      {
         console.log("error is: "+ error);
      }
    });
});
}
```

Listing. 2. Excerpt CREATE function.

ut-SUT receives and parses the triggering message to invoke the `generateOperation` with all of its parameters on the IoT application. With respect to the triggering message received, where the parameter `op` is 1, the `generateOperation()` function triggers a create request on the IoT application. The `create` function is detailed below and is implemented by using JavaScript (https://developer.mozilla.org/bm/docs/Web/JavaScript).

When the TS receives this request, it evaluates its contents and sets the verdict of the test case that can be pass, fail, inconclusive, or error. This process continues in the loop to automatically execute all the test cases in the control file. Once all the test cases are executed, a final report is generated by the TS for assessing the degree of compliance of the

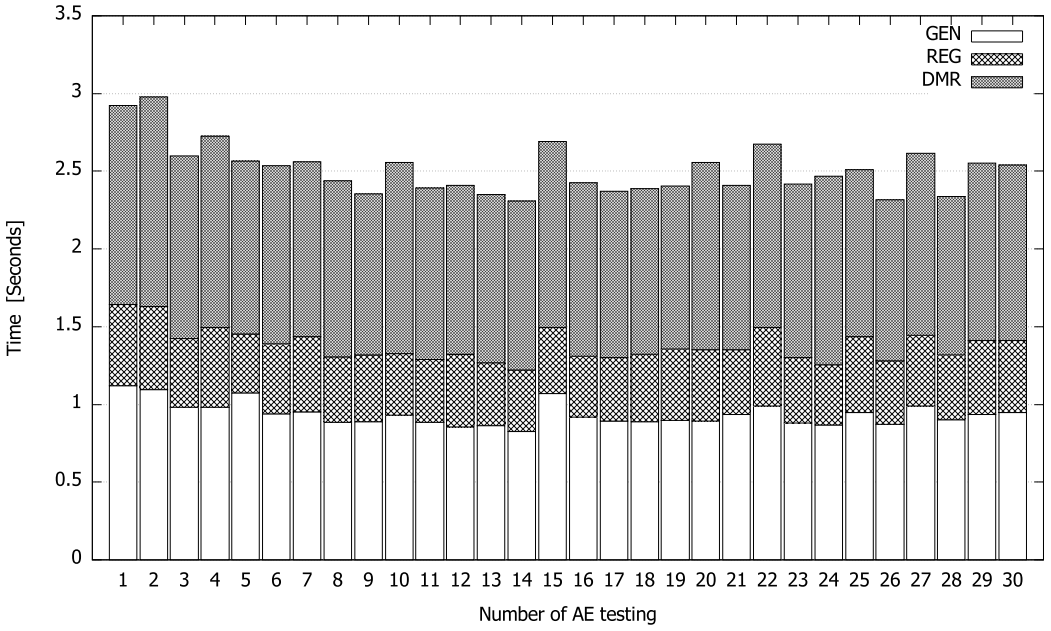

FIGURE 10. The oneM2M AE testing performance evaluation.

IoT application with the referring standards, in this case, the oneM2M specifications.

In addition, as shown in Fig. 10, we conducted a performance evaluation to show the feasibility and practicality of the proposed conformance testing mechanism. Conformance testing for AE was performed on a computer with UBUNTU 16.04 LTS OS, an Intel Core i7-7800X CPU @ 3.50 GHz X2 processor, and 8.0 GB of memory. For the testing, we prepared a total of sixty test cases for general IoT service layer capability (20 test cases), registration of IoT devices (9 test cases), data management-related features (27 test cases), and management of repository functions (4 test cases). Such test cases were developed by the oneM2M Test Working Group as sections of testing specifications. We participated and contributed to the development of the test cases together with other testing experts from industry. For each test case, we performed AE conformance testing 30 times. It took 2.5 seconds to complete one test run for all sixty test cases, and the results showed a standard deviation of 0.165. Fig. 11 shows the GUI of the ut-TS embedded testing system after executing a set of test cases.

In order to see the performance advantage of AUTOCON-IoT, we conducted conformance testing for an IoT application with three different testing configurations, i.e., manual testing, hybrid testing with dedicated stimulus, and AUTOCON-IoT. When the IoT application conformance testing was performed based on the automated testing approach as shown in Table 2, it took 3.071 seconds to complete one test set, while stimulus tool-based hybrid testing and manual testing took approximately 113.9 and 274.2 seconds, respectively, to complete one test set.

The reason why automated testing performed better than manual and hybrid testing is that all procedures were automated, such as setting the testing parameters and the order of sending and receiving messages between the TS and the SUT so that all test cases could be verified in one execution. In other words, the method using automated testing minimized the tester's intervention, thereby reducing errors in performing testing.

TABLE 2. oneM2M IoT application conformance testing results.

Testing Profile	Operation	Items	AUTOCON-IoT	Hybrid-testing	Manual testing
GEN	CREATE	6	0.359	11.3	30.4
	RETRIEVE	6	0.306	12.7	27.8
	UPDATE	6	0.195	9.4	28.7
	DELETE	6	0.261	10.5	29.6
REG	CREATE	8	0.466	14.4	31.4
	DELETE	1	0.205	3.5	30.2
DMR	CREATE	13	0.782	19.7	54.2
	RETRIEVE	7	0.267	14.2	27.8
	UPDATE	4	0.117	8.8	21.5
	DELETE	3	0.113	9.4	19.6
Total	-	60	3.071 (s)	113.9 (s)	274.2 (s)

In the case of hybrid testing that uses dedicated software tools with predefined stimulus testers do not need to develop the tools from scratch, and they can reduce the redundant time by using tools' various support functions such as drop-down testing menu, crafting test messages. However, radical problems still remain. To synchronize testing sessions between a testing system and target IoT application, at least two testers have to be involved in. In addition, if a way to test the IoT applications is modified by an organization managing test specifications, these changes have to be reflected as soon as possible; however, it is not an easy task. As described in Table 2, by using such tools, testers could get slightly better results than the one from manual testing. However, unless human involvement is fully replaced with standardized

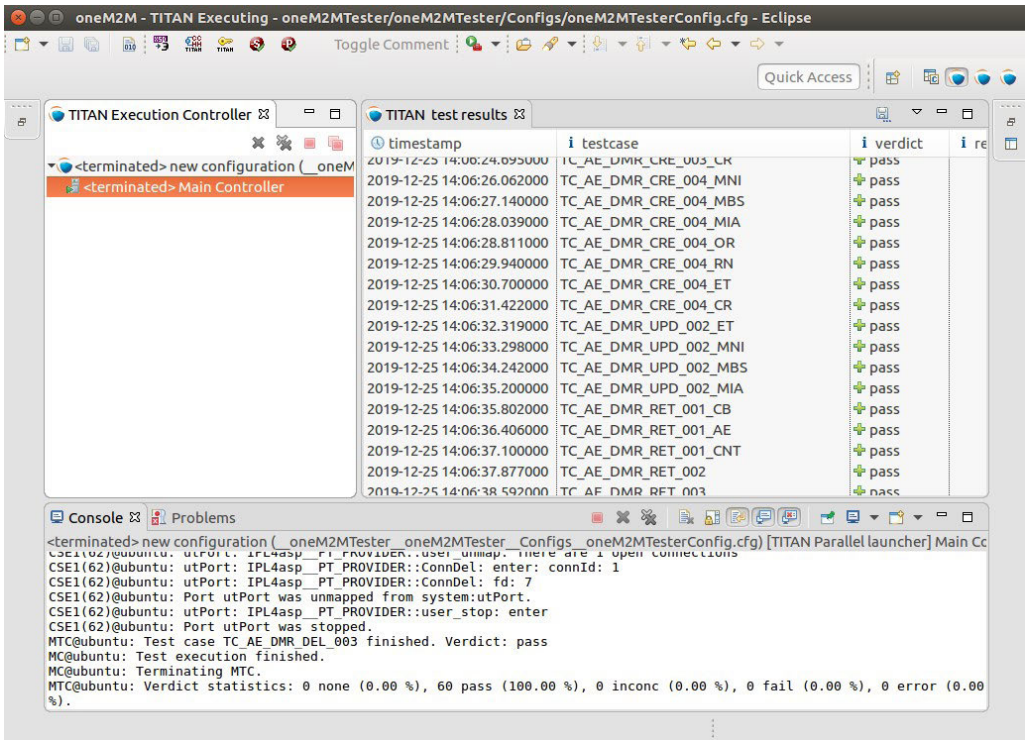

FIGURE 11. A screen capture of performing an automated testing with an ut-ST embedded oneM2MTester.

triggering message, it is not feasible to test large number of IoT applications.

The oneM2M Test Working Group annually hosts two INTEROPS events that offer test sessions to vendors and developers to assess the level of interoperability of their implementations and verify the correct interpretation of the oneM2M specifications. In particular, in the last oneM2M INTEROPS 5 & 6, which were held in Dec. 2017 in Pangyo, South Korea, and Jul. 2018 in Washington DC, USA, respectively, oneM2MTester with the proposed ut-ST feature was introduced to run conformance testing to help them debug their products as well as evaluate the feasibility of the oneM2MTester. The oneM2MTester conducted conformance testing with more than 15 implementations and provided various detected debugging issues that were fixed by developers during the events.

V. RELATED WORK

In [38], it briefly explained the concept of a testing framework for IoT by integrating the concept of simulation, unit integration and end-to-end integration testing. In addition, [39] explored compliance of IoT devices with privacy policy agreements and established models regarding the privacy criteria to measure the degree of compliance. However, those approaches have the disadvantages that both testing systems and test experts must be physically located at the same place to test their products.

Accordingly, the cloud-based testing system has been a great alternative for dealing with the previous issue. The characteristics of cloud computing can enhance service delivery [40], production cost and time reduction [41] and responsiveness towards requirement changes. In this regard, cloud computing can be used as an IoT testing platform remotely supporting IoT testing, logs and results views. The F-Interop [42], a cloud-based interoperability testing framework, provides a testing expert remote testing environment supporting a variety of IoT standards and protocols and standards. However, it heavily concentrates on the interoperability aspect. [43] developed TTCN-3-based testing suites called IoT-Testware to support the widely used IoT protocols from the conformance testing perspective, but its coverage is limited to message queueing telemetry transport (MQTT) and constrained application protocol (CoAP). The IoT compatibility testing tool (ICAT) [44] was designed to support remote IoT testing, but it only examines compatibility issues of IoT devices on the level of firmware.

In addition, there are several research works for improving the problems of manual testing. Dobles *et al.* [45] compared the effectiveness of automated and manual tests in terms of total test time and the number and severity of defects found. The results showed that automated testing is more effective than manual testing at finding defects. To make the test script automatically, a graphic user interface (GUI)-based testing component was proposed to define and implement the test scripts automatically in a large industry system [46]. With the considerable attention on artificial intelligence (AI) technologies, a natural language-based automated testing approach was studied [47]. The proposed mechanism shows that a manual testing script written in English can be almost automatically converted to mechanical interpretation. In conclusion,

much previous research has attempted to support IoT testing and automated testing. However, as far as we know, standardized conformance testing for a large number of IoT applications in constraint devices is not well considered.

VI. CONCLUSION

The paper describes an automated and scalable conformance testing mechanism by introducing a triggering mechanism in the current state of the art of conformance testing in the IoT. We noted some of the challenges of conformance testing in the IoT. The current IoT market is fragmented due to the inefficiency in conformance testing, which creates interoperability issues between multiple IoT applications. Due to the large scale of the IoT applications, which each have dozens of features, it is difficult to perform manual conformance testing. Our approach for automatic conformance testing helps to speed up the process of testing and certification for the IoT standardization bodies. By performing the automatic conformance testing process, we can assure the interoperability of the multiple implementations of the same standard without actually setting up interoperability testing between each implementation individually. As oneM2MTester was successfully introduced in the oneM2M INTEROP events and produced positive results by speeding up the process with no errors by reducing the human intervention, we plan to use this approach in further interoperability events and improve it according to the demands and further requirements.

Moreover, this approach is compatible with remote testing that has the test system in a cloud service. For remote testing, automated testing is a required mechanism. Remote testing activities can reduce the costs for individuals in gathering and performing testing in the same environment. This approach for automatic and scalable conformance testing is truly a key factor and must have mechanisms to improve IoT standardization and interoperability. As a future work, we consider to extend our configuration to fully distributed IoT environment [48], [49] and develop a testing solution to perform conformance testing in various places such as edge, fog and cloud.

REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.
[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2347–2376, 4th Quart., 2015.
[3] D. Evans, "The Internet of Things: How the next evolution of the Internet is changing everything," CISCO, San Jose, CA, USA, White Paper, 2011, vol. 1, pp. 1–11.
[4] M. Blockstrand, T. Holm, L.-Ö. Kling, R. Skog, and B. Wallin, "More than 50 billion connected devices," Ericsson, Stockholm, Sweden, White Paper, 2011, vol. 14, p. 124.
[5] S. Ziegler, C. Crettaz, L. Ladid, S. Krco, B. Pokric, A. F. Skarmeta, A. Jara, W. Kastner, and M. Jung, "IoT6—Moving to an IPv6-based future IoT," in *The Future Internet*. Berlin, Germany: Springer, 2013, pp. 161–172.
[6] B. Kang and H. Choo, "An experimental study of a reliable IoT gateway," *ICT Exp.*, vol. 4, no. 3, pp. 130–133, Sep. 2018.
[7] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed Internet of Things," *Comput. Netw.*, vol. 57, no. 10, pp. 2266–2279, Jul. 2013.
[8] R. Khan, S. U. Khan, R. Zaheer, and S. Khan, "Future Internet: The Internet of Things architecture, possible applications and key challenges," in *Proc. 10th Int. Conf. Frontiers Inf. Technol.*, Dec. 2012, pp. 257–260.
[9] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, "Internet of Things: Vision, applications and research challenges," *Ad Hoc Netw.*, vol. 10, no. 7, pp. 1497–1516, Sep. 2012.
[10] R. S. Sinha, Y. Wei, and S.-H. Hwang, "A survey on LPWA technology: LoRa and NB-IoT," *ICT Exp.*, vol. 3, no. 1, pp. 14–21, Mar. 2017.
[11] D. Bandyopadhyay and J. Sen, "Internet of Things: Applications and challenges in technology and standardization," *Wireless Pers. Commun.*, vol. 58, no. 1, pp. 49–69, Apr. 2011.
[12] J. Song, A. Kunz, M. Schmidt, and P. Szczytowski, "Connecting and managing M2M devices in the future Internet," *Mobile Netw. Appl.*, vol. 19, no. 1, pp. 4–17, Nov. 2013.
[13] A. Vallejo, J. Ruiz, J. Abella, A. Zaballos, and J. M. Selga, "State of the art of IPv6 conformance and interoperability testing," *IEEE Commun. Mag.*, vol. 45, no. 10, pp. 140–146, Oct. 2007.
[14] S. Seol, M. Kim, S. Kang, and J. Ryu, "Fully automated interoperability test suite derivation for communication protocols," *Comput. Netw.*, vol. 43, no. 6, pp. 735–759, Dec. 2003.
[15] S. Maag and C. Grepet, "Interoperability testing of a MANET routing protocol using a node self-similarity approach," in *Proc. ACM Symp. Appl. Comput. (SAC)*, 2008, pp. 1908–1912.
[16] R. Hao, D. Lee, R. K. Sinha, and N. Griffeth, "Integrated system interoperability testing with applications to VoIP," *IEEE/ACM Trans. Netw.*, vol. 12, no. 5, pp. 823–836, Oct. 2004.
[17] M. Schmidt, A. Wilde, A. Schulke, and H. Costa, "IMS interoperability and conformance aspects [IP multimedia systems (IMS) infrastructure and services]," *IEEE Commun. Mag.*, vol. 45, no. 3, pp. 138–142, Mar. 2007.
[18] B. S. Ahmed, M. Bures, K. Frajtak, and T. Cerny, "Aspects of quality in Internet of Things (IoT) solutions: A systematic mapping study," *IEEE Access*, vol. 7, pp. 13758–13780, 2019.
[19] B. Sand, "IoT testing—The big challenge why, what and how," in *Proc. Int. Internet Things Summit*. Berlin, Germany: Springer, 2015, pp. 70–76.
[20] D. Rayner, "OSI conformance testing," *Comput. Netw. ISDN Syst.*, vol. 14, no. 1, pp. 79–98, Jan. 1987.
[21] M. Krichen and S. Tripakis, "Conformance testing for real-time systems," *Formal Methods Syst. Des.*, vol. 34, no. 3, pp. 238–304, 2009.
[22] M. Krichen and S. Tripakis, "Black-box conformance testing for real-time systems," in *Proc. Int. SPIN Workshop Model Checking Softw.* Berlin, Germany: Springer, 2004, pp. 109–126.
[23] W. E. Howden, "Functional program testing," *IEEE Trans. Softw. Eng.*, vol. SE-6, no. 2, pp. 162–169, Mar. 1980.
[24] S. Moseley, S. Randall, and A. Wiles, "Experience within ETSI of the combined roles of conformance testing and interoperability testing," in *Proc. 33rd Eur. Solid-State Device Res. (ESSDERC)*, Oct. 2003, pp. 177–189.
[25] S. Moseley, S. Randall, and A. Wiles, "Experience within ETSI of the combined roles of conformance testing and interoperability testing," in *Proc. 33rd Eur. Solid-State Device Res. (ESSDERC)*, 2003, pp. 177–189. [Online]. Available: https://ieeexplore.ieee.org/document/1251206
[26] R. G. de Vries and J. Tretmans, "Towards formal test purposes," in *Proc. Formal Approaches Test. Softw. (FATES)*, vol. 1, 2001, pp. 61–76.
[27] *TTCN-3 Core Language*, ETSI No. ES 201 873-1, Version 4.11.1, Apr. 2019. [Online]. Available: http://www.ttcn-3.org/index.php/downloads/standards
[28] C. Willcock, T. Dei, S. Tobies, S. Keil, F. Engler, and S. Schulz, *An Introduction to TTCN-3*, vol. 2. Hoboken, NJ, USA: Wiley, 2005.
[29] J. Grabowski, D. Hogrefe, G. Réthy, I. Schieferdecker, A. Wiles, and C. Willcock, "An introduction to the testing and test control notation (TTCN-3)," *Comput. Netw.*, vol. 42, no. 3, pp. 375–403, Jun. 2003.
[30] J. Z. Szabó and T. Csöndes, "Titan, TTCN-3 test execution environment," *Infocommun. J.*, vol. 62, no. 1, pp. 27–31, 2007.
[31] S. Husain, A. Prasad, A. Kunz, A. Papageorgiou, and J. Song, "Recent trends in standards related to the Internet of Things and machine-to-machine communications," *J. Inf. Commun. Converg. Eng.*, vol. 12, no. 4, pp. 228–236, Dec. 2014.
[32] J. Swetina, G. Lu, P. Jacobs, F. Ennesser, and J. Song, "Toward a standardized common M2M service layer platform: Introduction to oneM2M," *IEEE Wireless Commun.*, vol. 21, no. 3, pp. 20–26, Jun. 2014.
[33] *Testing Framework, oneM2M Std. V2.0.0*, Standard TS-0015, 2016. [Online]. Available: http://www.onem2m.org/technical/published-documents
[34] *Implementation Conformance Statements, oneM2M Std.*, Standard TS-0017, 2017. [Online]. Available: http://www.onem2m.org/technical/latest-drafts

[35] *Test Suite Structure and Test Purposes, oneM2M Std.*, Standard TS-0018, 2017. [Online]. Available: http://www.onem2m.org/technical/latest-drafts
[36] *Abstract Test Suite Implementation Extra Information for Test, oneM2M Std.*, Standard TS-0019, 2017. [Online]. Available: http://www.onem2m.org/technical/latest-drafts
[37] W. F. Goncalves, C. B. de Almeida, L. L. de Araujo, M. S. Ferraz, R. B. Xandu, and I. de Farias, "The influence of human factors on the software testing process: The impact of these factors on the software testing process," in *Proc. 12th Iberian Conf. Inf. Syst. Technol. (CISTI)*, Jun. 2017, pp. 1–6.
[38] M. Bures, "Framework for integration testing of IoT solutions," in *Proc. Int. Conf. Comput. Sci. Comput. Intell. (CSCI)*, Dec. 2017, pp. 1838–1839.
[39] A. Subahi and G. Theodorakopoulos, "Ensuring compliance of IoT devices with their privacy policy agreement," in *Proc. IEEE 6th Int. Conf. Future Internet Things Cloud (FiCloud)*, Aug. 2018, pp. 100–107.
[40] L. Riungu-Kalliosaari, O. Taipale, and K. Smolander, "Testing in the cloud: Exploring the practice," *IEEE Softw.*, vol. 29, no. 2, pp. 46–51, Mar. 2012.
[41] J. Gao, X. Bai, W.-T. Tsai, and T. Uehara, "Testing as a service (TaaS) on clouds," in *Proc. IEEE 7th Int. Symp. Service-Oriented Syst. Eng.*, Mar. 2013, pp. 212–223.
[42] M. R. Palattella, F. Sismondi, T. Chang, L. Baron, M. Vučinić, P. Modernell, X. Vilajosana, and T. Watteyne, "F-Interop platform and tools: Validating IoT implementations faster," in *Proc. Int. Conf. Ad-Hoc Netw. Wireless*. Cham, Switzerland: Springer, 2018, pp. 332–343.
[43] I. Schieferdecker, S. Kretzschmann, A. Rennoch, and M. Wagner, "IoT-testware—An Eclipse project," in *Proc. IEEE Int. Conf. Softw. Qual., Rel. Secur. (QRS)*, Jul. 2017, pp. 1–8.
[44] W.-K. Chen *et al.*, "ICAT: An IoT device compatibility testing tool," in *Proc. IEEE 25th Asia–Pacific Softw. Eng. Conf. (APSEC)*, 2018, pp. 668–672.
[45] I. Dobles, A. Martinez, and C. Quesada-Lopez, "Comparing the effort and effectiveness of automated and manual tests," in *Proc. 14th Iberian Conf. Inf. Syst. Technol. (CISTI)*, Jun. 2019, pp. 1–6.
[46] C. Klammer and R. Ramler, "A journey from manual testing to automated test generation in an industry project," in *Proc. IEEE Int. Conf. Softw. Qual., Rel. Secur. Companion (QRS-C)*, Jul. 2017, pp. 591–592.
[47] S. Thummalapenta, S. Sinha, N. Singhania, and S. Chandra, "Automating test automation," in *Proc. 34th Int. Conf. Softw. Eng. (ICSE)*, Jun. 2012, pp. 881–891.
[48] D. Korzun, A. Varfolomeyev, A. Shabaev, and V. Kuznetsov, "On dependability of smart applications within edge-centric and fog computing paradigms," in *Proc. IEEE 9th Int. Conf. Dependable Syst., Services Technol. (DESSERT)*, May 2018, pp. 502–507.
[49] D. Kyriazis and T. Varvarigou, "Smart, autonomous and reliable Internet of Things," in *Proc. 1st Int. Workshop Commun. Sensors Netw. (ComSense), 4th Int. Conf. Emerg. Ubiquitous Syst. Pervasive Netw. (EUSPN)*, Niagara Falls, ON, USA, Oct. 2013

JAEYOUNG HWANG received the bachelor's degree in computer engineering from Sejong University, Seoul, South Korea, in 2015, where he is currently pursuing the Ph.D. degree. He is a member of the Software Engineering and Security Lab (SESLab) and also working as an Assistant Researcher at the IoT Platform Research Center of Korea Electronics Technology Institute (KETI). He contributed a lot in the oneM2MTester tool, which is the conformance testing tool to support the one M2M standard.

ABDULLAH AZIZ received the B.S. degree from the University of the Punjab, Pakistan, and the M.S. degree from Sejong University. He is currently pursuing the Ph.D. degree with Luleå Tekniska Universitet.

NAKMYOUNG SUNG received the B.S. and M.S. degrees in electronic and information engineering from the Hankuk University of Foreign Studies, South Korea, in 2010 and 2015, respectively. He is currently a Senior Researcher with the Autonomous IoT Research Center, KETI. His research interests include the Internet of Things, embedded system software, future internet architecture, and standard for M2M/IoT device systems.

ABBAS AHMAD is currently a Researcher with the Université de Franche-Comté. To conclude his master's degree, he has accomplished a six-month internship at the Research and Development Department, Airbus Operations SAS, where he developed Isami Analyst Testing Tool (IATT), a non-regression test tool. His Ph.D. entitled "Model-Based Testing for IoT System, Methods & Tools" was realized at Easy Global Market in collaboration with the University of Bourgogne Franche Comté. He also has an International Software Testing Qualifications Board (ISTQB) foundation level certification, an ISTQB Model-Based Tester foundation, a FIWARE developer's week Brussels Diploma, and a TTCN-3 Certificate in test automation. With a strong knowledge of software and application testing, he also has been a FIWARE friendly tester for four months, where he tested the FIWARE cloud portal and some Generic Enablers. One of his achievements is winning the first prize of the Internet of Things Hackathon during the IoT Week (Lisbon 2015). During his PhD studies and working at EGM, he has been involved in more than five innovative FP7/H2020 European Commission Project and has participated to several Specialist Task Force (STF) at the European Telecommunications Standards Institute (ETSI).

FRANCK LE GALL is currently the CTO of Easy Global Market, an innovative SME focused on integration and validation of emerging technologies. He is driving company development of advanced testing technologies as well as the integration of IoT and data platforms. He involves himself in the standardization area, including oneM2M and FIWARE. Previously, he has participated in large Research and Development projects within the big industries, i.e., Orange, Alcatel, Thomson, and has spent nine years as the Director of an innovation management company. He has directed more than 10 large scale projects and studies related to the evaluation and monitoring of innovation and technical programs as well as research projects. He is currently participating in several EU funded projects.

JAESEUNG SONG (Senior Member, IEEE) received the B.S. and M.S. degrees in computer science from Sogang University, and the Ph.D. degree from the Department of Computing, Imperial College London, U.K. He worked for NEC Europe Ltd., from 2012 and 2013, as a leading Standard Senior Researcher. From 2002 to 2008, he worked for LG Electronics as a Senior Researcher, leading a 3GPP SA Standard Team. He is currently an Associate Professor, leading the Software Engineering and Security Lab (SESLab), Computer and Information Security Department, Sejong University. His research interests span the areas of software engineering, software testing, networked systems and security, with a focus on the design and engineering of reliable IoT/M2M platforms, particularly in the context of the semantic IoT data interoperability, secure software patch techniques, the blockchain IoT, and edge computing. He holds the position of the *IEEE Communications Standards Magazine* IoT Series Editor, oneM2M Technical Plenary Vice-Chair, and the TTA IoT/M2M Convergence Special Project Group Chair.