Correct-by-Construction Approach for Formal Verification of IoT Architecture

Zinah Hussein Toman, Lazhar Hamel, Sarah Hussein Toman, Mohamed Graiet

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 207 (2022) 2598–2609

Procedia Computer Science

www.elsevier.com/locate/procedia

26th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES 2022)

Correct-by-Construction Approach for Formal Verification of IoT Architecture

Zinah Hussein Toman [a,b*], Lazhar Hamel[c], Sarah Hussein Toman[a,b] Mohamed Graiet[d]

[a] Department of Computer Science, FSM, Al-Monastir University, Tunisia

[b] Department of Computer Science, Faculty of Computer Science and Information Technology, Al-Qadisiyah University, Iraq

[c] Department of Computer Science, ISIMM, Al-Monastir University, Tunisia

[d] Department of Computer Science, ENSAI, Rennes, France

Abstract

Nowadays, The Internet of Things(IoT) has shown an increased interest in the academic literature, while its implementations became involved in almost every aspect of life in modern society. IoT is the integration of virtual and physical things through distributed services to collect and share data among themselves. The number of architecture approaches designed to aid IoT has increased significantly recently. As a result of different IoT architecture approaches, in this paper, we proposed a novel correct-by-construction formal approach based on an Event-B method to describe the physical architecture of IoT layers. This formal approach inspects four layers: the physical layer, the gateway layer, the middleware layer, respectively the application layer. An Electrocardiogram (ECG) IoT system is applied in our model as a case study. Finally, we proved and validated the correctness of our formal model by using proof obligations and the model checking tool called Rodin.

Keywords: IoT architecture, Event-B, Internet of Things(IoT), Formal model

1. INTRODUCTION

In the new global emerging world, IoT technologies have become an increasing interest for many governments, organizations, and companies [1].IoT incorporates various kinds of services, communication protocols, and hardware. The Internet of Things is the collaboration of objects through the Internet using various communication technologies,

* Corresponding author.
Email address: zinah.hussein@qu.edu.iq

10.1016/j.procs.2022.09.318

sensors, and actuators[2].Different architecture-based layers are proposed by different authors, however, they all share the same basic concept, the IoT layer uses the bottom-to-top approach. Sensors collect data using smart devices, controllers process data and make decisions, then communicate with persons or devices[3]. The applications or users are in the upper layer, the technology specifies addresses to smart devices and the communication media is utilized in the lower layer. Software such as controllers are used to processes the data, which belong to the midst layer[4].

Formal methods represent a useful tool for using mathematics and logic for verification and specification of the accuracy of designs. In the IoT field, formal methods are used to perform many approaches that are related to our lives. Also, it makes it possible to prepare security guarantees with respect and real-time properties to a given model. When the proper design is established, the correct code can be generated or implemented from the design[5]. However, there is a lack of research to verify the IoT systems in terms of the physical architecture of IoT layers. Therefore, this paper presents a novel formal model of IoT architecture layers based on an Event-B method. This model is developed incrementally from the abstract level to the target level by using the refinement mechanism. At each refinement, we modeled the layers starting with the physical layer, followed by the gateway layer, middleware layer, and application layer respectively. To validate the functionality and the correctness of the IoT layers model, we used an Electrocardiogram(ECG)IoT system in our model and the Rodin model checker and proof abdications. The paper is structured as follows: Section2 introduces other works that are related to our approach;Section3 describes the concept and architecture of IoT and gives an outline of Event-B theories;Section4 proposes a motivating example;Section5 presents the formalization of IoT architecture in the Event-B model;Section6 proposes a verification and validation of our approach and Section7 concludes this paper.

2. RELATED WORK

The study of IoT technology has been devoted a great deal of effort For several years. However, most of the contributions are focusing on the IoT architecture. Topics of Formal methods are still very inadequate, fragmented, and largely focused on only a few aspects of this domain.This sub-section presents a literature review of some of these works organized chronologically in order.

The work featured in[6]is dealing with the wireless sensor–actor networks, which are present in both typical sensor nodes and more developed and strong nodes, called actors. In this paper, the authors present various, formal models for those types of wireless networks. Recently, formulas of those models' have been presented with an algorithm for returning actor–actor coordinate links by the existing sensor infrastructure. In[7],the authors present the physical layer protocol, its first timed automata model, and use automatic verification to demonstrate fault tolerance under hardware assumptions and several error models.They are using the real-time model checker UPPAAL to evaluate several correctness properties and repair values for the model parameters. The work presented in[8] shows a solution for supporting the deployment and design of IoT applications.They use formal verification techniques to guarantee the correct construction of the composition of objects and suggest a dependable deployment mechanism for an IoT application. It is an interface-based model that combines a behavioral specification for the IoT object.They provide a general view of the device and its behavior.

In[9], the authors suggest a unified approach through a framework in Event-B for verifying IoT Communication protocols.They start with an abstract model of these protocols that include common features of different protocols.Then the model is refined into concrete models for IoT protocols using decomposition and refinement techniques of Event-B.

The work presented in[10]displays a solution for the management and development of smart city services by using formal methods to develop safe and secure systems.The authors suggest an approach that employs Event-B theories to minimize the proof efforts and support the process of data refinement for structuring Event-B models. After that, the proposed approach verifies the smart city system based on reuse modeling.

In[11],a formal method approach of IoT Conflict Checker is presented to ensure the safety of controller and actuators' behavior with respect to conflicts, where any policy violation will be identified. This approach defines the safety policies for actions, controllers, and triggering events. After that, the proposed approach proves logical soundness and completeness by using Prolog. In[12],the authors propose a general framework for the formal depiction of the physical architecture and control software of an IoT-based system. The proposed framework approaches analysis and modeling of the invariant architectural properties, then can add specific properties progressively and check them. The suggested framework is using the Event-B model to experiment and implement. Compared to these

works, while there is a lack of research to verify the IoT systems that deal with the physical architecture of IoT layers, our approach introduces a novel formal verification approach based on an Event-B method to describe the physical architecture of IoT layers.

3. BACKGROUND

3.1. 3.1 IoT concept

Currently, the term IoT is a trend.The term was coined for the first time in 1999 by Kevin Ashton working at Procter & Gamble, to describe his approach to introduce RFID tags and sensors on products in the supply chain, which has obtained a lot of attention in academia and industry since then[3].

The primary concern of IoT is the paradigm of connecting things without human intervention through a network and enables the exchange and collection of information across the network. This is performed by connected devices and “things” across some technologies to the Internet and other networks. These technologies include machine-to-machine(M2M),wireless sensor network(WSN),automated teller machine(ATM),and other connected devices[13].Although the development of technology has changed the definitions of things, a key aspect of it has linked many objects through the means of their embedded actuators, and the sensors can sense the environment of the system and collect information, then it is passed on to the next IoT layer,or sometimes gateways or network layer using one of the protocols.These objects not only interacting with a physical world and gather information from the environment, but also provide services for analytics, information transfer, communication and applications using existing Internet standards[14].

3.2. 3.2 IOT architecture

In simple terms, every IoT deployment capable of serving its designed purpose needs a strong IoT architecture. The applicability of the system and resulting efficiency largely depends on the quality of the infrastructure developed. Until this time, there has been no standard architecture for IoT. Nevertheless, the authors proposed different architectures for IoT, such as[15].The first basic model is the three-layer architecture that covers perception, networking, and application layer; the four-layer architecture(Middleware-based architecture)which is composed of perception, networking, middleware, and application layer. The function of the middleware layer involves data storage, service management, service composition[16];and a five-layer architecture. In this section, we briefly described the generic architecture of five layers[17,18,19,20,21]see Fig. 1.

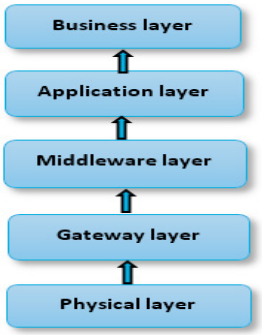

Fig 1. The IoT architecture of five layers

- Physical layer: also known as perception layer, the component of this layer is represented by sensors, actuators, and devices. In this layer, sensing technologies(e.g.,sensor,TAGs,GPS,etc.)are used to perceive the physical properties of the IoT environment. It converts the physical information which is collected by sensing technologies, into digital signals, which is a convenient form for communication and transmission. We also can

use embedded sensors when some objects might not be perceived. So, the embedded sensors are beneficial technologies used to facilitate works of perception layer by processing the data at the end devices.

- Transport layer or Access gateway layer: transfer the sensor data that is gathered by sensors or devices in the perception layer using wired or wireless communication channels to the middleware layer and vice versa.

- Middleware layer or Processing layer: provides crucial functionalities.It stores, processes, aggregates, filters, and analyses a huge amount of data that is received from the access gateway layer.This layer can also manage and provide access control to the devices and perform discovery.

- Application layer: accounts for delivering application services that come from the middleware layer to different applications.For example,smart health, smart home, and smart cities.

- Business layer: responsible for managing the IoT system. It does decision-making analysis from data, that derives information. The success of any service in an IoT system does not depend on only processes and technologies used in it,but also on how they are presented to consumers. The Business layer is implementing these tasks for the IoT system. It involves making graphs, flowcharts, how the device can be improved, analysis of results,etc.

To build an effective and robust IoT formal model, initially,we derive the right architectural requirements for each layers.The functional requirements that we will use in our formal model are listed in the following steps:

- Phy1:Each thing has a name, type, location, and properties.

- Phy 2:Physical layer components are IoT sensors, actuators, and devices.

- Phy3:Each sensor has data.

- Phy4:In the physical layer we can add data from any IoT device.

- Phy5:In the physical layer we can add one or more sensors.

- Phy6:In the physical layer we can add one or more actuators.

- Phy7:Physical layer collect data from a sensor or other devices.

- Gat8:The gateway layer responsible of sending the collected data from the sensor or other device to the controller by some technologies.

- Gat9:Each controller has data.

- Mid10:The controller software is responsible of processing and analysis the data which came from the previous layer, then produce orders to actuators or devices according to specific rules.

- Mid11:Any actuator receives data from the controller only.

- Mid12:The orders of controllers represent the input of the devices or actuators.

- App13:The actuator’s output sends a signal to the device to change the state of the device and take action.

- App14:The device’s output is making action to the environment.

3.3. Event-B

Event-B is a modeling method for developing and formalizing systems whose components can be modeled as discrete transition systems. Development of the classical B, also called B-method[22],Event-B is now centered on the general concept of events. The main advantage of Event-B is that it offers a refinement process that allows the complexity of a system to be mastered[23].The Event-B model is constructed from two basic elements:context and machine.The context describes the static part of modeling and specifies the parameters of a formal model and its properties which have the constants, carrier sets, theorems and axioms. A machine describes the dynamic part of a model and defines a set of variables, theorems, invariants, variants, events, respectively a set of guarded events[24]. See Fig.2.

The main feature of Event-B is that it allows models to be evolved incrementally through mechanisms such as context extension and machine refinement. These technologies enable users to evolve systems from their abstract specifications and thereafter provide more implementation details. More importantly, the properties are maintained through refinement that is proved at the abstract level, and therefore are also ensured to be satisfied by later refinements. As a result, correctness proofs of systems are divided and distributed among various levels of abstraction, which is easier to manage. Rodin[25]is the platform that introduces effective modeling, refinement, and proof of Event-B models. And offers many features, such as the theory feature.

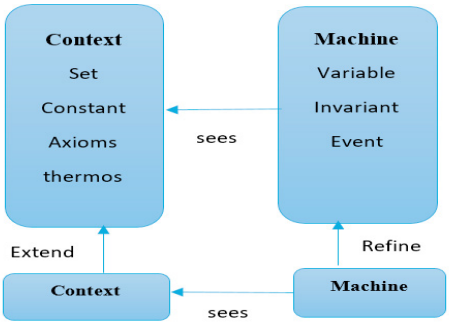

Fig.2. An Event-B Project Structure

4. CASE STUDY

In this section, we introduce an IOT system for Electrocardiogram(ECG)monitoring as a case study to clarify our model.(see Fig3).

The Electrocardiogram(ECG)monitoring contains four components, each component represents a thing in

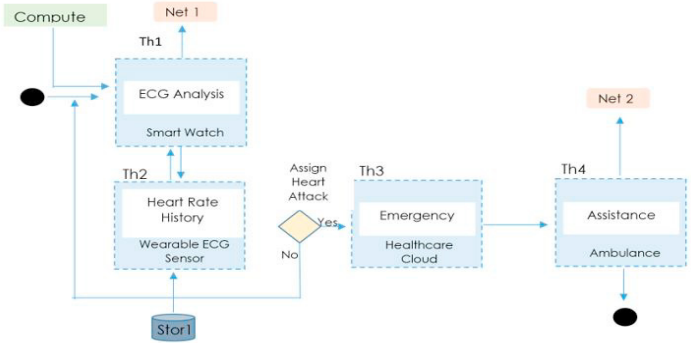

Fig.3. An overview of the Electrocardiogram (ECG) system

the IoT system: Smartwatch(Th1) Wearable ECG Sensor(Th2),Healthcare Cloud(Th3) and Ambulance (Th4). The workflow of the Electrocardiogram(ECG) monitoring according to each layer in our model is: to get the last sensor reading the ECG Analysis“Smartwatch”(Th1)that periodically passing the control to Heart Rate History“Wearable ECG Sensor“(Th2). Then after returns the control from Heart Rate History to ECG Analysis. If there are signs of a heart attack, then pass the control from ECG Analysis to the Emergency service “Healthcare Cloud“(Th3),which sends the control to the Assistance service“Ambulance“(Th4) to locate the nearest available ambulance.Finally, the control is returned the ECG Analysis. The four layers of IoT architecture are explained in this case study.All four components which represent things are included in the physical layer. The process of passing control from Smartwatch to Wearable ECG Sensor and vice versa and from Smartwatch to Healthcare Cloud are included in the Gateway layer. The middleware layer includes all processes of analysis and processing data. The software that interacts with users by Smartwatch is a part of the application layer.

5. FORMALIZATION IOT ARCHITECTURE IN EVENT-B

In this section, our approach is presented to formalize the physical architecture of IoT layers. The IoT architecture formal model consists of an abstract model and three refinements as illustrated in Fig.4. We start by modeling the physical layer in the abstract model.After that, we extend it to the first refinement to model the gateway layer, followed by the second refinement to model the middleware layer. Finally, the third refinement will concern the modeling of the application layer.

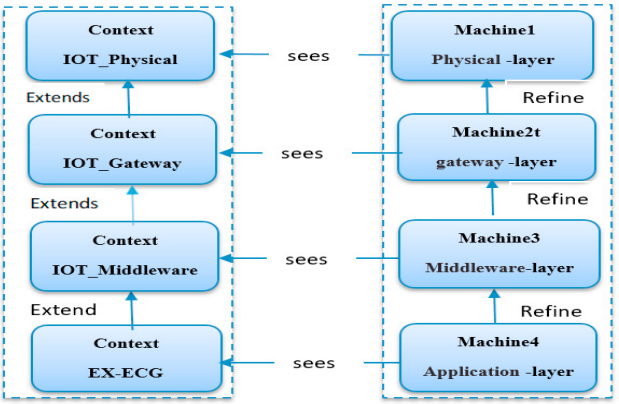

Fig.4. An overview of an Event-B formal approach of the architecture of IoT layers

5.1 Abstract model

This section presents the detailed process of constructing the abstract model.The abstract model represents the structure of the physical layer. It consists of context IOT_physical and the machine Physical layer as detailed below:

Context: The first context in the Event-B model for IoT Architecture is IOT_physical. It consists of a collection of sets(clause SETS)that represents the description of different concepts of an abstract model, such as defining the things as thing, thing name, thing type, thing properties, and thing location. The fields of variables concerning these types of data are thing, nameset, typeset, proset, locatset and the description of containing physical layer such as sensors,

```
CONTEXT
        IOT_physical
SETS
        thing ,nameset,
        typeset, proset, locatset, SENSOR,
        ACTUATOR, DEVICE,DATA
AXIOMS
        axm1  :  finite(thing)
        axm2  :  finite(ACTUATOR)
        axm3  :  finite(SENSOR)
```

Fig.5. An Event-B model for IoT Architecture: the context (1)

actuators, devices, and data as set name SENSOR, ACTUATOR, DEVICE, DATA.The AXIOMS depicts the properties of the attributes determined in the SETS. In this context, the axm1, axm2, axm3 are added to specify that all the defined sets thing, SENSOR, ACTUATOR respectively are finite. See Fig.5.

Machine: The machine Physical layer responsible for the most concepts defined in the physical layer that contains a set of things, where each thing has a physical component represented by sensors, actuators, and devices(that satisfy the Phy1 and Phy2).The main function of this layer is to collect data from the IoT environment. In another word, the machine is sees(clause SEES)in the above context IOT_physical that represents the first machine in the Event-B model for IoT Architecture as illustrated in Fig.6. Then we defined some variables to model the components of the abstract model. However,the inv1, inv2, inv3, and inv4 is typing invariants which model the Phy1 that represents each thing, has a name, type , properties, and location respectively. Like this, inv5, inv6, and inv7 are used to model the Phy2. This machine defines the data variable as a subset of DATA set (invariants inv8). Finally, the invariant inv9 models the sensor that contained data Phy3.

The addData is an event in the abstract model for adding data from any device in IoT environment, where any d (parameter)is included in DATA set(grd1)and not included in data set(grd2),then added to data set(act1). Similar to this event, the events addSensor and addActuator are adding one or more sensors and actuators, respectively which satisfied Phy4,Phy5, and Phy6 as illustrated in Fig.7.

```
MACHINE
        physical layer
SEES
        IOT_physical
VARIABLES
        thinglocat,
        thingname,
        thingpro,
        thingtype,
        sensor
        ,actuator,
        data,
        data_sensor,
        DEV
INVARIANTS
        inv1  :  thingname ∈ thing ↔ nameset
        inv2  :  thingtype ∈ thing ↔ typeset
        inv3  :  thingpro ∈ thing ↔ proset
        inv4  :  thinglocat ∈ thing ↔ locatset
        inv5  :  sensor ⊆ SENSOR
        inv6  :  actuator ⊆ ACTUATOR
        inv7  :  DEV ⊆ DEVICE
        inv8  :  data ⊆ DATA
        inv9  :  data_sensor ∈ sensor ↔ data
```

Fig.6. An Event-B model for IoT Architecture: The machine

```
addData
 STATUS
   ordinary
ANY
  d
WHERE
    grd1 : d ∈ DATA
    grd2 : d ∉ data
THEN
    act1 : data ≔ data ∪ {d}
END
addSensor
                                    STATUS
    ordinary
  ANY
  s
 WHERE
    grd1 : s ∈ SENSOR
    grd2 : s ∉ sensor
 THEN
    act1 : sensor ≔ sensor ∪ {s}
END
addActuator
 STATUS
ordinary
  ANY
  ac
 WHERE
    grd1 : ac ∈ ACTUATOR
    grd2 : ac ∉ actuator
```

Fig. 7. The events of adding data, sensor, and actuator for IoT Architecture.

The main function in the physical layer for IoT Architecture is to collect data from IoT environment by sensors or other devices. The event collectDataSensor indicates this function and modeled Phy7 See Fig.8.

```
collectDataSensor ≙
STATUS
 ordinary
ANY
 d
 s
WHERE
 grd1  :  d ∈ data
 grd2  :  s ∈ sensor
 grd3  :  s∉dom(data_sensor)∧d∉ran(data_sensor)
THEN
 act1  :  data_sensor ≔ data_sensor ∪ {s ↦ d}
END

END
```

Figure.8. The collectDataSensor event for IoT Architecture

5.1. First Refinement

In IoT architecture-based layer, the second layer is the Gateway layer that is responsible for transferring the data collected in the physical layer to the controller(Next layer) which satisfied Gat8.This layer is modeled in the first refinement for the Physical layer. The set of the existing controllers are modeled by the variable controller which is a subset of the CONTROLLER set(invariants inv10).The invariants inv11 modeled Gat9 that each controller(controller)has data (data_controller) See Fig.9.

An event named dataTocontroller is defined to model the transferred data set (grd3) where any d (parameter) is included in data and any data_sensor collected from sensor Set(grd4)is transferred to controller Set(act1) as illustrated in Fig.10.

```
inv10  :  controller ⊆ CONTROLLER
inv11  :  data_controller ∈ controller ↔ data
```

Figure.9. An Event-B model for IoT Architecture: The machine (2)

```
dataTocontroller ≙
STATUS
 ordinary
ANY
 d
 c
 s
WHERE
 grd1  :  c ∈ controller
 grd2  :  s ∈ sensor
 grd3  :  d ∈ data
 grd4  :  s ↦ d ∈ data_sensor
 grd5  :  c ↦ d ∉ data_controller
 grd6  :  c∉dom(data_controller)∧d∉ran(data_controller)
THEN
 act1  :  data_controller≔ data_controller ∪ {c↦d}
END

END
```

Figure.10. The dataTocontroller event for IoT

5.2 Second Refinement

This section introduces the second refinement of the model in IoT architecture representing a third layer(Middleware layer) which receives data from the Gateway layer, then processes and analyzes it to produce orders and send it to the IoT sensors or devices. With this aim in mind, we model the function of the middleware layer that defines a context name IoT_middleware which is extended to IoT_Gateway that contains SET and CONSTANTS with their properties to model this layer. The axioms axm1 define the constants (on,off) to determine the state of a device(see Fig.11. The machine Middleware layer performs its work layer which models the requirement Mid10 ,See Fig.12.

```
CONTEXT
    IOT_Middleware  ›
EXTENDS
◦    IOT_Gateway
SETS
◦   ORDER    ›order send from controller
◦   STATE    ›state of device on or off
CONSTANTS
◦   on   not symbolic ›
◦   off  not symbolic ›
AXIOMS
◦   axm4:   partition (STATE, {on}, {off})
END
```

Fig.11. An Event-B model for IoT Architecture the middleware context

```
inv13   :   (data_actuator) ⊆ ran
            (data_controller)
inv14   :   order_dev ⊆ ORDER
inv15   :   dev_in ⊆ order_dev
inv16   :   dev_out ⊆ order_dev
inv17   :   order_actu ⊆ ORDER
inv18   :   actu_in ⊆ order_actu
inv19   :   actu_out ⊂ order_actu
inv20   :   compute ∈ data ↔ ORDER
inv21   :   c_order ⊆ ORDER      //
```

Fig.12. An Event-B model for IoT Architecture: The machine (3)

However, the event computeOrder defines a compute function by using data data_controller that exists in controller set(grd3)and produces orders set act1(see Figure13).An event named controllerToactuator and controllerTodevice they are defined to model the sending of order from the controller to the actuators and devices, respectively. This models the requirements Mid11 and Mid12 see Fig.14, Fig.15.

```
computeOrder   ≙
STATUS
 ordinary
ANY
 d      //
 c      //    subset of controller
WHERE
 grd1   :   d ∈ data
 grd2   :   c ↦ d ∈ data_controller
 grd3   :   c∉dom(data_controller)∧d∉ran(data_controller)
THEN
 act1   :   c_order ≔ c_order ∪ {compute      //
            (d)}
END
```

Fig.13. The computeOrder event for IoT Architecture.

```
controllerToactuator   ≙
STATUS
 ordinary
ANY
 ord_in
WHERE
 grd4   :   ord_in ∈ c_order
 grd6   :   ord_in ∉ dev_in
 grd7   :   ord_in ∉ order_actu
THEN
 act1   :   order_actu ≔ order_actu ∪ {ord_in}
END
```

Fig.14. The controllerToactuator event for IoT Architecture.

```
controllerTodevice   ≙
STATUS
 ordinary
ANY
 ord_in
WHERE
 grd1   :   ord_in ∈ c_order
 grd4   :   ord_in ∉ order_dev
THEN
 act1   :   order_dev ≔ order_dev ∪ {ord_in}
END

END
```

Fig.15. The controllerTodevice event for IoT Architecture.

5.3 Third refinement

This refinement is extended to formalization developed in the previous refinement for the abstract model and is added to represent the application layer for delivering orders received from the previous layer and changing the state of a device to perform an action in the environment. In other words,the machine Applications layer has one event deviceAction for checking and changing the state of the device set act1 and modeling the requirements App13 and App14 ,See Fig.16.

```
deviceAction   ≙
STATUS
 ordinary
ANY
 dev
WHERE
 grd2   :   dev ∈ DEV
 grd3   :   state_D(dev) = off
THEN
 act2   :   state_D(dev):= on
END

END
```

Fig.16. The deviceAction event for IoT Architecture.

6. VERIFICATION AND VALIDATION

In this section, our validation is made up of two parts:ProB[26]and Proof obligations(POs)[27]. Firstly,ProB is an animator checker tool that allowed us to run an automatic animation to validate the correctness of the model specification step-by-step. By different values to the variables, carrier sets and constants, we animated the case study in our model. To apply this animation, initialization values have been given as illustrated in Fig.17.

By observing the different states, we can determine the correctness of the model behaver. By testing events of the model, ProB had no interruptions in any stage which indicates the fact that the model is behaviorally corrected. Finally, POs are generated by the Rodin platform to analyze the model then prove it. The analysis model regarding all invariants, events, and refinements to produce rule statements then proved them. The statement of POs rule is named as a sequent. Each sequent is represented as H ⊢ G, where H refers to a set of predicates named as hypotheses and G refers to Goal or conclusion. If the Hs are true, then G is true. In our model, 12 proof obligations have been discharged,

100% of proof obligations were discharged automatically by Rodin prover. Fig.18. shows the proof obligations of IoT Architecture formal model.

Fig .17. Initialization values of the Electrocardiogram (ECG) system

```
INITIALISATION
 STATUS
   ordinary
 BEGIN
 act1 : thinglocat ≔ {smartwatch ↦ locatT1, wearable ↦ locatT2,
   HC_cloude ↦ locatT3 ,ambulance ↦ locatT4 }
 act2 : thingpro ≔ {smartwatch ↦proT1, wearable ↦proT2,
   HC_cloude ↦ proT3 ,ambulance ↦proT4 }
 act3 : thingname ≔ {smartwatch ↦nameT1, wearable ↦nameT2,
   HC_cloude ↦nameT3 ,ambulance ↦nameT4 }
 act4 : thingtype ≔ {smartwatch ↦type1, wearable ↦type2,
   HC_cloude ↦type3 ,ambulance ↦type4 }
 act5 : sensor ≔{ s1,s2,s3,s4}
 act7 : actuator ≔{ AC1,AC2,AC3,AC4}
 act8 : order_actu ≔{actu_msg}
 act9 : actu_in ≔∅
 act10 : actu_out ≔∅
 act12 : order_dev ≔ {dev_msg}
 act13 : data ≔ {D1,D2,D3,D4}
 act14 : c_order ≔ {contr_msg}
 act15 : state_D ≔ {De1↦ on, De2↦ on,De3 ↦ on,De4 ↦ on}
 act16 : data_sensor ≔{s1 ↦ D1,s2 ↦ D2,s3 ↦ D3,s4 ↦ D4}
 act17 : data_actuator ≔{AC1 ↦ D1,AC2 ↦ D2,AC3 ↦ D3,AC4 ↦
   D4}
 act18 : dev_in ≔∅
 act19 : dev_out ≔∅
 act20 : controller ≔{contr1,contr2,contr3,contr4}
 act21 : data_controller ≔{contr1 ↦ D1,contr2 ↦ D2,contr3 ↦
   D3,contr4 ↦ D4}
 act22 : compute ≔{ D1↦ contr_msg, D2↦ dev_msg,
   D3↦dev_msg ,D4↦contr_msg}
 act23 : DEV ≔ {De1,De2,De3,De4}
END
```

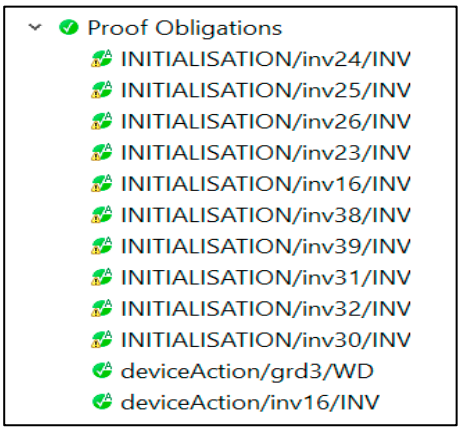

Figure.18. The proof obligations of IoT Architecture formal model.

7. CONCLUSION

Due to a lack of previous research to verify the IoT systems in terms of the physical architecture, this study presented a mathematical approach for designing and modeling the IoT architecture layers based on an Event-B method. A set of requirements is defined and taken into account in our model. Four layers are constructed in this model using the refinements mechanism: the physical layer, the gateway layer, the middleware layer and the application layer. The refinements allowed us to correct by constructing all IoT layers gradually and facilitating the proofs. An Electrocardiogram (ECG) IoT system was developed as a case study in our model. Finally, a validation of the correctness of our formal model is satisfied by using an animator checker tool called ProB and proof obligations.

References

[1] H. Ning and Z. Wang,“Future internet of things architecture:like mankind neural system or social organization framework?”IEEE Communications Letters,vol.15,no.4,pp.461–463,2011.

[2] M. Weyrich and C. Ebert,“Reference architectures for the internet of things,”IEEE Software,vol.33,no.1,pp.112–116,2016.

[3] J. Gubbi,R.Buyya,S.Marusic and M.Palaniswami,“Internet of Things(IoT):A vision, architectural elements,and future directions,”Future Generation Computer Systems,vol.29,issue7,2013,pp.1645–1660.

[4] Patel, M.,& Bhise,M.(2019,January).Raw data processing framework for IoT.In 2019 11th International Conference on Communication Systems &Networks(COMSNETS)(pp.695-699).IEEE.

[5] Wing,J. M.(1990).A specifier's introduction to formal methods.Computer,23(9),8-22.

[6] Kamali, M.,Liabnis,L., Petre,L.,&Sere,K.(2014).Formal development of wireless sensor-actor networks.Science of Computer Programming,80(2014),25–49.

[7] Gerke,M., Ehlers,R.,Finkbeiner,B.,&Peter,H. J.(2010, September).Model checking the flexray physical layer protocol.In International Workshop on Formal Methods for Industrial Critical Systems(pp.132-147).Springer, Berlin, Heidelberg.

[8] Krishna, A.,Le Pallec,M.,Mateescu,R.,Noirie,L.,&Salaün,G.(2019,May).Rigorous design and deployment of IoT applications.In 2019 IEEE/ACM 7th International Conference on Formal Methods in Software Engineering(FormaliSE)(pp.11-20).IEEE.

[9] Diwan, M.,&D'Souza,M.(2017,October).A framework for modeling and verifying IoT communication protocols.In International Symposium on Dependable Software Engineering:Theories,Tools,and Applications(pp.266-280).Springer,Cham.

[10] Alkhammash, E.H.(2021).Trustworthy smart city systems using refinement and Event-B Theories.Multimedia Tools and Applications,1-22.

[11] A. Al Farooq, E.Al-Shaer,T.Moyer,and K. Kant,"IoTC2:A formal method approach for detecting conflicts in large scale iot systems",In 2019IFIP/IEEE symposium on integrated network and service management(IM),pp 442-447,2019.

[12] Attiogbé, C.,&Rocheteau,J.(2019).Architectural Invariants and Correctness of IoT-based Systems.arXiv preprint arXiv:1912.08912.

[13] Barry H.,"Design,Launch,and Scale IoT Services:A Practical Business Approach" ,Apress,2018.

[14] Khan, R.,Khan,S.U.,Zaheer,R., & Khan,S.(2012, December).Future internet:the internet of things architecture,possible applications and key challenges.In 2012 10th international conference on frontiers of information technology(pp.257-260).IEEE.

[15] Pallavi S.,Smruti R.S.," Internet of Things: Architectures, Protocols, and Applications", Hindawi ,Journal of Electrical and Computer Engineering ,Volume2017,Article ID9324035,26 January2017.

[16] Ranganathan,A.,Al-Muhtadi,J.,Chetan,S.,Campbell R.,&Mickunas,M.D.(2004,October).Middlewhere: a middleware for location awareness in ubiquitous computing applications.In ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing(pp.397-416).Springer,Berlin,Heidelberg.

[17] Hammoudeh,M.;Epiphaniou,G.;Belguith,S.;Unal,D.;Adebisi,B.;Baker,T.;Kayes,A.;Watters,P.A service-oriented approach for sensing in the Internet of Things:intelligent transportation systems and privacy use cases.IEEE Sens.J.2020.

[18] Weyrich,M.,&Ebert,C.(2015).Reference architectures for the internet of things.IEEE Software,33(1),112-116.

[19] Zhang,A.L.(2016,June).Research on the architecture of internet of things applied in coal mine.In 2016 International Conference on Information System and Artificial Intelligence(ISAI)(pp.21-23).IEEE.

[20] Aazam,M.,Hung,P. P.,&Huh, E. N.(2014,April).Smart gateway based communication for cloud of things.In 2014IEEE ninth international conference on intelligent sensors,sensor networks and information processing(ISSNIP)(pp.1-6).IEEE.

[21] Wu,M.,Lu,T.J.,Ling,F.Y., Sun,J.,&Du,H.Y.(2010,August).Research on the architecture of Internet of Things.In 2010 3rd international conference on advanced computer theory and engineering(ICACTE)(Vol.5,pp.V5-484).IEEE.

[22] Hoang,T. S.(2013).An introduction to the Event-B modelling method.Industrial Deployment of System Engineering Methods,211-236.

[23] Gaaloul,W.,Bhiri,S.,&Rouached,M.(2010).Event-based design and runtime verification of composite service transactional behavior.IEEE transactions on services computing,3(1),32-45.

[24] Damchoom,K.,Butler,M.,&Abrial,J.R.(2008,October).Modelling and proof of a tree-structured file system in Event-B and Rodin. In International Conference on Formal Engineering Methods(pp.25-44).Springer,Berlin,Heidelberg.

[25] Abrial,J.R.,Butler,M., Hallerstede, S.,Hoang,T. S.,Mehta,F.,&Voisin,L.(2010).Rodin: an open toolset for modelling and reasoning in Event-B.International journal on software tools for technology transfer,12(6),447-466.

[26] Hoang,T.S.,"An introduction to the Event-B modelling method," Industrial Deployment of System Engineering Methods,2013,pp.211-236.

[27] M.Leuschel,and M.Butler,"ProB:A model checker for B,"In International symposium of formal methods europe(pp.855-874). Springer,Berlin,Heidelberg,2013.