Design and model checking of timed automata oriented architecture for Internet of thing

Guang Chen1,2,3,4, Tonghai Jiang1,5, Meng Wang1,3,4, Xinyu Tang1,3,4 and Wenfei Ji1,2,3,4

Abstract
The architecture model of the Internet of thing system is the primary foundation for the design and implementation of the Internet of thing system. This article discusses the method and practice of time automaton modeling and model checking for the architecture of the Internet of thing system from the state and time dimensions. This article introduces the theory and method of modeling using time automata. And then, combined with the actual need of the elderly health cabin Internet of thing system, a dynamic and fault-tolerant time automaton model is established through a relatively complete architecture modeling. The model checking method verifies that the designed Internet of thing system has no deadlock system activity, service correctness, and timeliness correctness. The results of modeling experiments and model validation show that the reference model of time automata Internet of thing architecture established in this article can better reflect the nature of interaction with the physical world, heterogeneity and large-scale, dynamic, and incompleteness of the Internet of thing system.

Keywords
Internet of thing, architecture, timed automata, modeling, model checking, UPPAAL

Date received: 19 November 2019; accepted: 7 February 2020
Handling Editor: Leo Zhang

Introduction
Internet of thing (IoT) refers to a dynamic loosely coupled system which combines various kinds of information sensing devices, such as radio frequency identification (RFID), sensing devices, and control devices that can execute certain commands to affect the environment through a variety of networking technologies and the Internet.1–4 Different from the traditional Internet, the IoT integrates physical space, information space, and social space.5 It has special characteristics different from the traditional Internet,6 including heterogeneity and large-scale, interaction with the physical world, resource constraints, dynamics, and incompleteness. These characteristics make the design and implementation of the IoT face great challenges, especially the lack of services caused by the dynamic and incomplete nature of intelligent devices.
Therefore, in the early stage of the design of the IoT system, it is necessary to establish a model of the IoT that accords with these characteristics, to formally verify and test the model on the basis of strict mathematics, and to ensure that the designed IoT system meets the expected design requirements and expected nature. The architecture of the IoTs is a graphical or formal model based on mathematics, formal language with precise semantics and specifications, or formal description method, which aims at the architecture of the IoTs from different perspectives and dimensions. Furthermore, the system architecture model which extracts the key features of the system from the specific IoTs system is further abstracted and generalized, so that it can provide reference for the design or modeling of other IoTs system, and the built IoTs architecture model is called the reference model of IoTs architecture.

In recent years, researchers in the field of IoTs have proposed a variety of architecture reference models for building the IoTs system, most of which are from the functional perspective. Researchers can model the composition and nature of the IoTs system from different perspectives and dimensions. The first “4 + 1” reference model proposed by the scholar Kruchten points out that the software architecture model is established from five different perspectives.

Because the IoT is a seamless integration of the physical world and the information world, with the continuous changes and uncertainties of the physical environment, service provided by the IoT system often has high dynamic and real-time requirements. Therefore, state and time attributes are one of the principal concerns of the IoT system. Intelligent devices in the IoT collect or affect the physical environment depending on a certain frequency and period, so the services provided by the IoT system should also have time- and state-related behavior.

In this article, the architecture of the IoT is modeled from the dimension of timed automata. It describes and reflects the time attributes and time constraints of the physical environment and the service behavior and state changes. In the process of modeling, the system characteristics of interaction between the IoT system and the physical world, heterogeneity and large-scale, dynamic, and incompleteness are embodied. And the service and timeliness correctness of the system model is verified by model testing. In this way, we discuss the method and practice of using timed automata to model the architecture of IoT.

### Related works

In the research of the architecture model of the IoT, in recent years, researchers in the field have proposed a variety of architecture reference models for building the IoT system. Mi et al. studied the implementation architecture of the social IoT. Shen et al. put forward a three-dimensional model of IoT architecture. In Dong-Yue et al. and Chen et al., the models of IoT system architecture proposed and established by various scholars were explained, respectively, from the functional perspectives, and a comprehensive and unified comparative analysis of these models was achieved.

The focus of these studies is to build an architecture model of the IoT from the dimension of system function and composition, and get the nature of the IoT system through theoretical analysis. None of these studies provides formal modeling and verification methods for architecture.

In this article, the architecture of the IoT is modeled from the dimension of timed automata. An easy but relatively complete timed automata architecture reference model, which reflects the modeling characteristics of the IoT, is established in section “Materials and methods.” Then in section “Results,” the model checking method is used to verify the system will not deadlock, the correctness of service, and the validity of the timeliness.

In theoretical research of time automata, the formal theory of time automata has been studied. The earliest theory of timed automata held that it is a four-tuple concept. Zhou et al. and Han et al. considered time automata as the concept of six-tuple in their theoretical research on time automata, but they did not consider the attribute variables and attribute constraints of time automata in the definition of the constraint set. At present, the precise formal definition of time automata theory has not been finalized in academia, and concepts related to time automata are not perfect in the formal theoretical research.

In this article, based on Bengtsson Johan’s theory of time automata, combined with the modeling practice of time automata, the six-tuple formal definition of time automata is proposed in section “Modeling theory and method of timed automata.” We give precise formal definitions of timed automata, timed automata status and status transformation, and timed automata network. Based on these theoretical studies, it explains how to use timed automata to build the architecture model of the IoTs system.

In terms of modeling and model checking using timed automata, Li et al. established a framework for the provision of IoT services. Ting et al. and Meng et al. have modeled and analyzed timed automata for robot systems and key communication nodes. Xiang and Xiao-Chun applied the method of time automaton modeling and model checking to the temporary speed limit of high-speed rail transponder. Liu et al. modeled and studied a cyber physical system (CPS) home lighting control system, but did not give a clear modeling process and model detection method. Xian-
Li and Chu-Jiao modeled and analyzed timed automata such as RFID and other common IoT terminal sensing devices, but it did not connect them with the cloud service, and modeled the complete IoT system. Li et al. combined with an application case of an intelligent conference room, gives a relatively complete modeling process, but the sensors and other control devices will fall into failure, and the considerations of perception and judgment of these failures are insufficient.

All the above research studies focus more on timely response to external physical environment, but do not reflect the effect and influence of the designed timed automata system on the physical environment. And none of the above studies considers the heterogeneity and large-scale of the IoTs system. In addition, these studies do not consider the dynamic and incomplete of the IoT system sufficiently in the process of modeling, do not consider the possible failure situation, and do not provide the corresponding fault-tolerant mechanism in the modeling of resources of the IoT system.

In this article, the characteristics of IoT system are fully considered in the modeling process of the elderly health cabin temperature control IoT system. In section “Modeling of air physical environment,” the physical environment timed automaton model reflects the interaction between the IoT system and the physical environment. In section “Modeling of temperature sensor,” aiming at the heterogeneity and large-scale characteristic of the IoT system, we propose a solution to build a timed automata template component library. In section “Modeling practice of IoT system,” taking the failure mode of the IoT system resources as an example, we discuss the way to embody the dynamic and incomplete characteristic using the timed automata to model the IoT architecture.

Materials and methods

This section first introduces the modeling theory and method of timed automata, gives the formal definition of timed automata, and the formal definition of status transformation and timed automata network. Then, based on the theoretical research of timed automata, taking the health cabin temperature regulation system as an example, a relatively complete system modeling practice to build the model of state and time dimension of the IoT system is carried out.

Modeling theory and method of timed automata

Timed automata modeling of IoT is based on the formal theory of timed automata. On the basis of Bengtsson Johan’s timed automaton theory, we enrich and perfect the formalization theory of timed automata and give precise formal definitions of timed automata, timed automata status and status transformation, and timed automata network.

**Definition 1.** The formal definition of timed automata: A timed automaton A is a tuple of six

\[ A = (N, l_0, \Sigma, X, E, I) \]

where:

- \( N \) is a finite set of positions;
- \( l_0 \) is the initial location of timed automata, it is obviously that \( l_0 \in N; \)
- \( \Sigma \) is a set of messages. Element \( a \in \Sigma \) can represent the behavior of receiving and sending messages;
- \( X \) is a collection of clock attribute variables;
- \( E \subseteq N \times B(X) \times \Sigma \times 2^X \times N \) is a set of edges of timed automata, where \( 2^X \) represents the power set operation of the clock attribute variables set. Each element \( (l, g, a, r, l') \in E \) represents an edge that changes from position \( l \) to position \( l' \), which can be simply remembered as \( l \xrightarrow{a,r} l' \), where \( r \subseteq X \) represents a set of variables that need to be reset or assigned to complete this state transition. \( g \) represents the clock attribute constraints that need to be satisfied when completing this transformation. \( a \in \Sigma \) represents the message sent or received synchronously when this conversion is completed;

\[ I := N \rightarrow B(X) \]

is a mapping from the set of positions to the set of invariant clock constraints.

**Definition 2.** The definition of timed automata status:

The status of a time automaton is defined as a tuple of positions of the time automaton.

**Definition 3.** The status transformation of timed automata: For a pattern \( (l, u) \) of timed automata, if the
execution condition of pattern change \( \langle l, u \rangle \xrightarrow{g,u,e} \langle l', u' \rangle \) cannot be satisfied for any pattern change \( \langle l, u \rangle \xrightarrow{d} \langle l, u + d \rangle \) of time passing under this pattern, it is said that timed automata is deadlocked in the interpretation \( u \) of clock attribute. Deadlock is a transient concept. It must be explained for a specific clock attribute before the time automaton will enter the deadlock state.

**Definition 5.** Product of timed automata and timed automata network: For two timed automata \( A_1 = \langle N_1, I_{10}, \Sigma_1, X_1, E_1, I_1 \rangle \) and \( A_2 = \langle N_2, I_{20}, \Sigma_2, X_2, E_2, I_2 \rangle \), if their message sets satisfy \( \Sigma_1 \cap \Sigma_2 \neq \emptyset \), then the timed automaton \( A_1 \) and \( A_2 \) can be connected (assembled). The result of connection of \( A_1 \) and \( A_2 \) is called the product of these two timed automata, it is recorded as \( A_1 \parallel A_2 \).

The product of timed automata \( A_1 \) and \( A_2 \) can be regarded as a new timed automaton \( A_1 \parallel A_2 = \langle N_1 \cup N_2, \{ I_{10}, I_{20} \}, \Sigma_1 \cup \Sigma_2, X_1 \cup X_2, E_1 \cup E_2, I_1 \cup I_2 \rangle \). For a new timed automata \( A_1 \parallel A_2 \), if \( (\Sigma_1 \cup \Sigma_2) \cap \Sigma_i \neq \emptyset \) is satisfied, it can continue the operation of product. When two or more time automata are connected in series, result \( M = A_1 \parallel A_2 \ldots \parallel A_n \), if \( n \geq 2 \) is called timed automaton network.

The time automata modeling method of the IoT system is divided into two steps. First of all, each resource in the IoT system is modeled by timed automata through the way of analyzing the nature of the resource, fill in the set of contents specified in the timed automata formalization Definition 1, and get the timed automata model of the corresponding resources. And then, the timed automata model of the IoT resources is assembled by sending and receiving messages to form a timed automata network. Thus, we can get the timed automaton model of the whole IoT system.

To improve the efficiency of modeling, we use UPPAAL\(^{22}\) that developed jointly by Uppsala University in Sweden and Alborg University in Denmark as a modeling tool. Previous research studies\(^{23,24}\) point out that compared with other time automaton modeling tools, such as HyTech,\(^{25}\) Kronos,\(^{26}\) and Epsilon,\(^{27}\) UPPAAL has significantly higher performance in time and space than other three kinds of tools, and has significant efficiency. Detailed menu description and use method of UPPAAL modeling tool can be referred to research done by David.\(^{28}\)

**Modeling practice of IoT system**

In this section, we start from a practical requirement of IoT system modeling, and build a relatively complete state and time dimension architecture of IoT system. On the basis of the modeling theory and modeling method discussed previously, the physical interaction, heterogeneity and large-scale, dynamic, and incomplete nature of the IoT are fully considered, and a complete timed automata model with fault-tolerant mechanism is established.

The resources of the IoT system refer to all components of the IoTs system. This includes not only all kinds of intelligent devices and sensor devices with entities in the physical world, but also virtual computing services in the information world. The physical environment that IoTs system perceives and influences is also the concept category of IoTs system resources. From the structure chart of the IoTs system, the resources of the IoTs system can be divided into three categories.

Including the physical environment of the IoTs system, all kinds of intelligent devices in the physical environment sense and affect the state of physical environment. Because these devices are tangible in the IoTs system, these devices can be collectively referred to as the physical end entities. There are also various IoTs system services based on physical entity functions and cloud computing functions.

In the health cabin IoT system, three temperature sensors are connected to sense indoor temperature, an air-conditioning device is connected to refrigerator, and a heating device is connected to heating. All devices and sensors are scheduled by cloud control service. The system needs to ensure that the indoor temperature can reach the optimum temperature range of 26°C in an effective time.

This application case is simple, but it contains all the basic elements of the IoT system, and reflects the main characteristics of the IoT system. Therefore, it has good generalization ability and universal reference value.

**Modeling of air physical environment.** The physical environment model of healthy cabin includes indoor and outdoor temperatures. Both of them are variable. Only when the system is initialized, the specific indoor and outdoor temperatures can be determined. If the indoor temperature is less than the outdoor temperature, then the indoor temperature will increase by a certain increment (adjustable increment) with a certain time interval (adjustable time interval), but the maximum temperature will not exceed the outdoor temperature. On the contrary, if the indoor temperature is higher than the outdoor temperature, the indoor temperature decreases with a certain increment at a certain time interval. Indoor temperature can be sensed by temperature sensors. If the air-conditioning device starts to refrigerator, every certain time interval makes the indoor temperature drops by 1°C. If the heating starts, the temperature rises by 1°C at each interval.

When the actual system is deployed, the time interval and increment of temperature increase and decrease, and the time interval of temperature drop caused by air-conditioning refrigeration can be obtained by
experimental means, so as to establish and simulate the relative real air environment timed automata model, and improve the accuracy of modeling.

In UPPAAL, template functions can be utilized to deal with these variable parameters. The variable parameters used in system design are declared to the top of the air physical environment timed automata template, as showed in Figure 1. OUTSIDE_TEMPERATURE is used to represent the outdoor temperature of a healthy cabin, INIT_TEMPERATURE is used to represent the indoor temperature, INTERVAL is used to represent the temperature change interval, INC is used to represent the increase or decrease in temperature, and INCREMENT is used to represent the increment of each temperature change. Referring to the modeling method described in section “Modeling theory and method of timed automata,” the time automaton modeling of air physical environment is shown in Figure 1.

In order to prevent the air physical environment model from being deadlocked when the temperature is higher or lower than the outdoor temperature (OUTSIDE_TEMPERATURE), two status transformation edges are added to deal with this problem, as showed on the upper side of Figure 1.

The status transformation \( \langle \text{loop}, u \rangle \xrightarrow{\text{GetAirTemperature}} \langle \text{loop}, u' \rangle \) sends the current temperature of the air physical environment. This temperature will be received by the sensor model, which shows that the IoT system can perceive the physical environment.

When need to model a specific indoor air temperature model, we can quickly complete the time automata modeling for indoor temperature environment in different IoT system through the way of filling in the parameters in the UPPAAL’s model statement. As showed in Algorithm 1, the physical environment of air temperature in healthy cabins in summer and winter was modeled, respectively.

**Modeling of temperature sensor.** First, we analyze the function and behavior of the temperature sensor. A temperature sensor senses temperature once every certain time interval, and sends the sensed temperature to the cloud temperature sensor management service. If the temperature sensor does not complete temperature sensing within a specified time interval, it is considered that the temperature sensor has a malfunction. In accordance with the above modeling points of temperature sensor, we use UPPAAL to establish the timed automata template for the temperature sensor, as showed in Figure 2. Template parameter INTERVAL represents the time interval of temperature sensor sensing and sending temperature information.

We design all the temperature sensors that are connected to the IoT system can unconditionally enter the failure state, as showed in the \( \langle \text{run}, u \rangle \xrightarrow{\text{Fault}} \langle \text{fault}, u' \rangle \) status transformation in Figure 2. And these sensors can

---

**Figure 1.** Environment template of air in elderly health cabin.
be repaired artificially after failure, as showed in the \((r, a, u) \mapsto (r', a, u')\) status transformation in Figure 2, so as to reflect the dynamic nature of the IoT system. The timed automaton template of the temperature sensor can greatly simplify the work of modeling for temperature sensor timed automata, and solve the heterogeneous and large-scale problems of the IoT system modeling.

Although most of the resources of the IoT are heterogeneous, there are some similarities among these heterogeneous resources. Taking temperature sensor as an example, in the IoT system, although the type and class of the temperature sensor are different, its function of temperature sensing is the same. Therefore, we can set up a timed automata template for the temperature sensor. When we need to get the temperature sensor model with different working parameters, we can fill in the corresponding parameters, so as to solve the heterogeneity problem in the modeling of the timed automata in the IoT.

When establishing the timed automaton model of the IoT system entities, it may be necessary to face the situation of accessing multiple identical resources, such as the simultaneous access of three temperature sensors in the modeling demand of the health cabin. Through the template modeling function, we can create a number of resource models of the IoT system, and to some extent, deal with the large-scale problem when modeling the timed automata in the IoT. As showed in Algorithm 2, three different temperature sensors are created to simulate the simultaneous operation of three temperature sensors.

Furthermore, some commonly used resources of the IoT can be modeled to form a component library. There are pre-defined resource component templates in the component library. Building a model component library can greatly reduce the workload of IoT system modeling, so that the main work of modeling is

---

**Algorithm 1.** Create timed automaton template instance of IoT system resource

```plaintext
//Physical Environment in Summer and Winter
SummarAir=Air(50,35,100,true,1); // Outdoor 50°C, indoor 35°C, increases by 1°C per 100 time units
WinterAir=Air(0,15,100,false,1); // Outdoor 0°C, indoor 15°C, decreases by 1°C per 100 time units

// Temperature Sensors
Sensor1=Sensor(10); // Temperature sensing and sending every 10 time units
Sensor2=Sensor(10);
Sensor3=Sensor(10);

// Air conditioning
AirCon1=AirCon(20); // Temperature drops by 1°C per 20 time units under refrigeration
AirCon2=AirCon(20);

// Heating
Heating1=Heating(20); // Temperature rises by 1°C per 20 time units under heating

// Cloud control service
HouseManager=Manager(5,10); // Check whether the air conditioner or heating is fail every 5 time units

// Artificial Repair
HouseMan=Man(true,0,false,0); // All resource can fail at any time and cannot be repaired artificially
```

**Algorithm 2.** Assemble resources of health cabin IoT system to form a timed automata network

```plaintext
// Physical Environment —— Air
SummarAir,
// WinterAir,
// Perceptual physical entities —— three sensors
Sensor1,
Sensor2,
Sensor3,
// Regulatory entity —— Air conditioning
AirCon1,
AirCon2,
// Regulatory entity —— Heating
Heating1,
// Central cloud control service
HouseManager,
// Artificial Repair Service
HouseMan;
```
concentrated on the modeling of IoT system. When we need to use a template, we only need to load its working parameters according to the actual situation, and get the time automata model corresponding to the IoT system resources, so as to improve the efficiency of the modeling of the IoT system architecture.

**Artificial repair modeling.** In order to conveniently control the resource of the IoT to enter the fault state in a certain way and to deal with the failure of the system function caused by the resource failure of the IoT, the modeling of artificial repair is increased as shown in Figure 3.

Using the artificial repair model, we can control the sending and receiving of the Fault? signal through the variable parameter CAN_FAULT, so as to control whether the resources of the IoT system can enter the fault state. Boolean CAN_REPAIR is used to control whether the resources of the IoT system can be repaired when it is in failure state. At the same time, the time and frequency of resource failure and repair can be controlled by FAULT_INTERVAL and REPAIR_INTERVAL.

**Modeling of air-conditioning.** The model of the air-conditioning in the health cabin is shown in Figure 4. The air-conditioning can receive the AirConditionerOn? control signal to start refrigeration, and also can receive the AirConditionerOff? signal to turn off. In the refrigeration state, every INTERVA time unit has an impact on the air physical environment (sending AirCool signals).

When the air-conditioning is in normal condition, it can receive and respond to AirConditionerOn? and AirConditionerOff? messages, regardless of whether it is closed or refrigerated. If the air-conditioning fails, it cannot respond to any messages. In cloud control service, by initiating sending messages at a certain time and frequency, according to the response state of the air conditioner, we can judge whether it is in failure state. In the model shown in Figure 4, the transformation of AirConIsFault? is added to simulate that the air-conditioning cannot respond to any message.

**Modeling of heating.** The model of the heating in the health cabin is shown in Figure 5. The modeling of heating is very similar to air-conditioning. We still limit the heating system to fail, and repair them artificially by receiving RepairHeating? messages.

**Modeling of cloud control services.** In the health cabin, the cloud control service model is shown in Figure 6. In the central control service, there are three normal working states. The idel state indicates that the system is idle and does not manipulate any physical entities to affect the physical environment. Cool state indicates that the IoT system is controlling the air-conditioning to cool...
the environment. The heat state indicates that the system is controlling heating to heat the environment. The cloud control service also includes sensor_fault state for displaying sensor faults, aircon_fault state for displaying air-conditioning faults, and heat_fault state for displaying heating faults. Handle state is the intermediate state of control decision-making calculation based on temperature perception after the system perceives the environmental temperature. In addition, the cloud control service model also contains several temporary states for processing synchronization of multiple messages, which have been treated as atomic states without any status transformation of time change.

In the cool state or heat state, the cloud control service carries out an air-conditioning or heating operation checks every time unit of CHECK_INTERVAL. If there is not any response (it can only synchronize to AirConIsFault! or HeatingIsFault!), it enters the fault state of air-conditioning and heating, respectively. In the idel, cool, and heat state, the counting clock is started for timing. If the counting clock value exceeds SENSOR_INTERVAL (the sensing period of the sensor), then it is judged that the sensor has a malfunction and enters the sensor_fault state.

Different from the fact that the temperature sensor enters the fault state is unconditional, it needs to meet the guard constraint sensor.c>SENSOR_INTERVAL to enter the sensor_fault state from the idel state in the central control cloud service model. This is because the cloud temperature sensor management service modeling is different from the temperature sensor, and the behavior of the temperature sensor falling into the fault state has nothing to do with its clock counter. Therefore, in Figure 4, for the temperature sensor modeling, entering the fault state is unconditional. However, the cloud temperature sensor management service is different. Considering the implementation of the temperature sensor management service, we define a clock to time when the service starts to work. If we receive the message sent by the temperature sensor, we will reset the clock to zero. At the same time, a guard process is implemented to monitor the value of the clock counter in real time. When its count value is greater than SENSOR_INTERVAL, it shows that the temperature sensor is in fault state. Again, when the service enters the fault state, there must be a clock counter count result greater than SENSOR_INTERVAL. Therefore, sensor.c>SENSOR_INTERVAL is a necessary condition for the cloud temperature sensor service to enter the fault state, which should be more accurate as a guard constraint.

Assembly time automata network. In the UPPAAL model declaration, according to the resource composition of the elderly health cabin, the demand is set up, and the timed automata instance model of the physical environment, physical entity, and cloud control service is initialized, respectively, as showed in Algorithm 1.

In the model initialization of artificial repair, we specify that all resource in the IoT system can enter the fault state. By setting FAULT_INTERVAL to 0, we express the meaning that all resource can unconditionally enter the fault state at any time, so as to reflect the dynamic nature of the IoT system.
Even if the faulty sensor is connected to the system, or even not any sensor is connected, the cloud temperature sensor management service can quickly indicate the status of sensor_fault, which reflects the incompleteness nature of the IoT system. According to the demand, we create three identical sensor timed automata models, and design a cloud control service which will not enter a sensor_fault state as long as there is at least one temperature sensor is still working, so as to reflect the fault tolerance of the IoT system.

Finally, we assemble all the resource components timed automata models of the IoT system to form a timed automata network as showed in Figure 8. We get timed automata oriented architecture for the health cabin IoT system.

Results

In this section, we use UPPAAL verifier to verify the nature of the IoT system in the health cabin for the elderly. First of all, we give all the properties and its numbers discussed in this section

\[
\begin{align*}
E(\text{Sensor1 fault}) & \quad (1) \\
E(\text{AirCon1 fault}) & \quad (2) \\
E(\text{Heating1 fault}) & \quad (3) \\
A[] \text{not deadlock} & \quad (4) \\
A[] \text{SummarAir.temperature} \geq 26 & \quad (5) \\
A[] \text{WinterAir.temperature} \leq 26 & \quad (6) \\
A[] \text{not (AirCon1, cool and Heating1.heat)} & \quad (7) \\
A[] \text{HouseManger.airconFault imply AirCon1.fault} & \quad (8) \\
A[] \text{HouseManger.heatFault imply Heating1.fault} & \quad (9) \\
A[] \text{HouseManger.sensorFault imply (sensor1.fault and Sensor2.fault and Sensor3.fault)} & \quad (10) \\
A[] \text{not Sensor1.fault or not Sensor2.fault or not Sensor3.fault imply not HouseManager.sensorFault} & \quad (11) \\
A[] \text{not Sensor1.fault or not Sensor2.fault or not Sensor3.fault and not AirCon1.fault and SummarAir.temperature > 26 and Timer > 20 imply AirCon1.cool} & \quad (12) \\
A[] \text{not Sensor1.fault or not Sensor2.fault or not Sensor3.fault and not AirCon1.fault and Clock > 300 imply SummarAir.temperature \geq 26 and SummarAir.temperature \leq 27} & \quad (13)
\end{align*}
\]

Figure 7. Model checking for activity of IoT system in elderly health cabin.

Figure 8. Service correctness of elderly health cabin IoT system.
A½/C138 not Sensor 1: fault or not Sensor 2: fault or not Sensor 3: fault and not AirCon 1: fault and WinterAir: temperature < 26 and Timer > 20 imply Heating 1: heat

A½/C138 not Sensor 1: fault or not Sensor 2: fault or not Sensor 3: fault and not AirCon 1: fault and Clock > 300 imply WinterAir: temperature ≥ 25 and WinterAir: temperature < 26

Systematic activity of health cabin

In order to reflect the dynamic nature of the IoT system, we allow temperature sensors, air conditioners, and heating to fail. The corresponding UPPAAL verification statements are (1), (the other two sensors, Sensor 2 and Sensor 3, are identical, and they are no longer listed repeatedly) (2), and (3). Although physical entities may fail, the system will not fall into a deadlock state, and can operate steadily and normally, as described as (4).

Model verify results of the above four properties are shown in Figure 7, which proves that the system has no deadlock system activity.

Correctness of service in healthy cabin

We do not want any waste of resources in the designed for elderly health cabin. Therefore, in the IoT system, air-conditioning and heating will not be turned on at the same time. The formal description language of UPPAAL is described as (7). In the physical environment of summer, the temperature will never be lower than 26°C, while in the winter, the temperature will never be higher than 26°C, which is described as (5) and (6), respectively.

In order to satisfy the properties (5) and (6), it is necessary to ensure that the time interval of sensor acquisition temperature is less than the time interval of air-conditioning and heating affecting the air physical environment (in the models shown in Figures 2 and 4, the values of these two time intervals are 10 and 20, respectively, in the model initialization of Algorithm 1).

We hope that the cloud control service can accurately indicate that the air-conditioning and heating are in a failure state, which can be expressed as (8) and (9) using UPPAAL description language. In order to prove the fault tolerance of the IoT system, we hope that as long as the temperature sensor does not enter the fault state, the cloud control service will not show the sensor fault. If the cloud control service shows sensor fault state, all temperature sensors must be in failure state. UPPAAL verification rules are expressed as (10) and (11).

Model verify results for the properties (5)–(11) are shown in Figure 8. It proves that the elderly health cabin IoT system can operate effectively in both summer and winter physical environments, and there is no waste happened when working. Cloud control services can effectively sense and indicate the system failure.

Validity of timing in healthy cabin

Elderly health cabins can ensure the correctness of their services within an effective time frame. Even in the summer when the indoor temperature is as high as 35°C, when the temperature sensor and air-conditioning do not fail, the elderly health cabin IoT system can ensure that the air-conditioning can enter the refrigeration state in 20 time units and that the temperature can be reduced to 26°C–27°C in 300 time units, as shown in the properties (12) and (13). Even in the winter when the indoor temperature is below 15°C, when the temperature sensor and heating do not fail, the elderly health cabin IoT system can ensure that the heating can enter the heating state in 20 time units and that the temperature can be raised to 26°C–27°C in 300 time units, as shown in the properties (12) and (13).

The validation results of properties (12)–(15) are shown in Figure 9, which proves the timeliness of the design of the elderly health cabin IoT system.

Discussion

Different from the formal research studies establish the model of the IoT architecture from the dimension of system function and obtain the nature of the IoT system
through theoretical analysis, we model the IoT system from the dimension of state and time. This article presents a simple but relatively complete reference model of time automata for the health cabin IoTs system, and tests the model based on the formal theory of time automata from the aspects of system activity, service correctness, and timeliness. It provides a good reference for the design and model validation of the IoT system using the formal modeling method of time automata and model detection technology.

This article starts from the characteristics of the IoTs system itself, and considers fully the interactivity with physical world, heterogeneity and large-scale, dynamic, and incomplete characteristics of the IoTs system in the process of modeling.

In terms of the interaction between the IoTs and the physical world, previous modeling practices have not separately modeled the timed automata for the physical environment in which the system is located. Li et al. considered that physical environment is a perceived entity in the modeling of IoT system using time automata, and devices such as air conditioners and fluorescent lamps used to regulate the state of the physical environment are controlled entities. This article holds that air-conditioning and fluorescent lamps are physical entities, and the corresponding entities should be temperature sensors and photometric sensors, rather than physical environment temperature and photometry. Physical environment is an invisible objective reality, not a concept of an entity, and the physical environment itself is not an integral part of the IoTs system. Therefore, it should not be classified into air-conditioning and other equipment of the IoTs system.

The IoTs system is the integration of the physical world and the information world. When modeling the architecture of the IoT system from the dimension of state and time, we must consider the real-time perception and effective control of the state of the physical environment in which the IoT system works. Therefore, it is necessary to build a separate time automaton model for the physical environment. The environmental state information that can be perceived by the IoT system is perceived from the physical environment model. The operation and work of the IoT system control equipment will also affect the state change of the physical environment time automata model, thus constituting an effective closed-loop timed automata network of the IoT system.

This article establishes a separate model of the physical environment of a healthy cabin in section “Materials and methods.” And in section “Results,” the model validation proves that the health cabin IoT system can correctly perceive (acquire) temperature information from the physical environment and affect the physical environment in a certain way, such as 300 time units to make the temperature reach 26°C.

In terms of heterogeneity and large-scale, no consideration has been given in previous research studies. Although this article does not directly simulate the access of hundreds of sensors, researchers can easily create any number of time automaton models through declaration of timed automata resource template. For example, on the basis of temperature sensor template, we can declare any number of temperature sensor timed automata models in the declaration of Algorithm 2, and we can configure different sampling intervals for each sensor model. In addition, this article also points out the method of building a common resource component library of IoTs system to better deal with the heterogeneous resources and large-scale access of the IoTs system.

In terms of dynamics and incompleteness, previous research studies do not consider the possible failure of the accessed devices. Most of the resources do not design failure states, and there is no design for resource failure awareness and identification in the perception service. This article allows all physical entities connected to the IoT system, including temperature sensors, air conditioners, and heating, to enter the failure state unconditionally. The method of model checking in section “Results” proves that cloud control services can effectively perceive and indicate the working status of all resources, whether accessing multiple device or not.

In this article, the main characteristics of the IoTs system are fully taken into account when modeling the time automata system architecture of IoT system. An example model of the time automata in a fault-tolerant healthy cabin is modeled, which can be used as a typical case for the application of time automata in the modeling of the IoTs system.

Conclusion

The architecture of the IoT is a graphical or formal model of the composition and working principle of the IoT system from different perspectives and dimensions. It is the primary basis for the design and implementation of the IoT system. This article comparatively explores the modeling method and practice of time automata in state and time dimension architecture of IoT.

In order to make better use of the timed automata architecture model of the IoT, the future research can study how to transform the clock attribute constraint and status transformation described in the timed automata into the coding method in software engineering, and then realize the automatic assembly and the direct generation of system service code of the IoT system resources under the support of component operation platform and code generation tool. In this way, we can
narrow the distance between the design of IoT system architecture and the implementation of IoT system, ensure that the implementation of IoT system strictly follows the principles of architecture design, improve the efficiency of the construction of IoT system, and reduce the implementation cost of IoT system.

Future research work can also be based on the actual construction need of the IoT system to establish a more dimensional description of the IoT architecture, for example, assembly architecture of the components of the IoT system, the entity deployment architecture of the IoT established in the system deployment phase, and the operation architecture of the maintenance and evolution phase of the IoT system, and to study its precise architecture modeling method and model detection method based on mathematical and formal theory. Through this way to ensure that the ultimate implementation of the IoT system meets the desired nature and need.

Declaration of conflicting interests
The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding
The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was partially supported by Wuxi science and Technology Development Fund Project (N20191005).

ORCID iD
Guang Chen https://orcid.org/0000-0002-4456-5981

References
1. Chen H-M and Cui L. Design and model checking of service oriented software architecture for Internet of things. Chin J Comput 2016; 39(5): 853–871.
2. Dong-Yue L, Gang Y and Bo Q. Summary of research on Internet of things architecture. Comput Sci 2018: 45: 27–31.
3. Yang L, Jin-Shen S and Yun H. Summary of Internet of things module technology and industry development. Design Techn Posts and Telecommun 2017; 9: 19–22.
4. Sun X, Zhang H, Meng W, et al. Primary resonance analysis and vibration suppression for the harmonically excited nonlinear suspension system using a pair of symmetric viscoelastic buffers. Nonlin Dyn 2018; 94: 1243–1265.
5. Mi B-T, Liang X and Zhang S-S. A survey on social Internet of things. Chin J Comput 2018; 41(7): 1448–1475.
6. Kruchten P. The 4 + 1 view model of architecture. IEEE Software 1995; 12: 42–50.
7. Chen H-M, Cui L and Xie K-B. A comparative study on architectures and implementation methodologies of Internet of things. Chin J Comput 2013; 36(1): 168–188.
8. Zhang R-H, He Z-C, Wang H-W, et al. Study on self-tuning tyre friction control for developing main-servo loop integrated chassis control system. IEEE Access 2017; 5: 6649–6660.
9. Shen S-B, Mao Y-Q, Fan Q-L, et al. The concept model and architecture of the Internet of things. J Nanjing Univ Post Telecommun 2010; 30(4): 1–8.
10. Zhou Q-L, Ji L-X and Wang Y-M. Model checking of real-time systems based on UPPAAL. Comput Appl 2004; 9: 129–131.
11. Li-Fang Z. Formal analysis and application of real-time systems based on UPPAAL and UML. Suzhou, China: Suzhou University, 2008.
12. Liu M, Ma W, Deng S, et al. Modeling and verification of services oriented cyber physical systems. J Comput Appl 2017; 34(6): 1770–1773.
13. Li L-X, Jin Z and Li G. Modeling and verifying services of Internet of things based on timed automata. Chin J Comput 2011; 34(8): 1365–1377.
14. Han D-S, Yang Q-L and Xing J-C. UML-based modeling and formal verification for software self-adaptation. J Software 2015; 26(4): 730–746.
15. Hua G, Lei Z and Xiyong Z. UPPAAL—a tool suit for automatic verification of real-time system. Microcomput Inform 2006; 15: 52–54 + 190.
16. Bengtsson J and Yi W. Timed automata: semantics, algorithms and tools. In: Desel J, Reisig W and Rozenberg G (eds) Lectures on concurrency and petri nets. Berlin: Springer, 2003, pp.87–124.
17. Le X. Research and implementation of real-time system modeling and formal verification based on UML and UPPAAL. Hangzhou, China: Hangzhou University of Electronic Science and Technology, 2017.
18. Ting L, Rui W, Xiao-Juan L, et al. Modular modeling and verification of industrial robots. Comput Eng Design 2019; 40(1): 209–217.
19. Meng Y, Li X, Guan Y, et al. Modeling and verification of robot joint bus communication system. J Software 2018; 29(6): 1699–1751.
20. Xiang Z and Xiao-Chun W. Modeling and verification of temporary speed limitation for high-speed railway cross-border based on MSC and UPPAAL. Railway Stand Design 2016; 60(10): 126–131.
21. Xian-Li J and Chu-Jiao C. Verification of accuracy of RFID positioning system based on UPPAAL. Comput Techn Develop 2016; 26(9): 104–108 + 113.
22. Behrmann G, David A, Larsten KG, et al. Uppaal 4.0. In. Proceedings of international conference on the quantitative evaluation of systems, Washington, DC, 16 October 2006. Washington, DC: IEEE Computer Society.
23. Yan X, Li Y and Li X. Real-time simulation of automotive systems based on UPPAAL. In: Proceedings of the 8th IEEE international conference on software engineering and service science (ICSESS), Beijing, China, 24–26 November 2017, p.4. New York: IEEE.
24. Xiong H, Zhu X and Zhang R. Energy recovery strategy numerical simulation for dual axle drive pure electric
vehicle based on motor loss model and big data calculation. *Complexity* 2018; 2018: 4071743.

25. Henzinger TA, Ho P-H and Wong-Toi H. HyTech: a model checker for hybrid systems. In: *Proceedings of international conference on computer aided verification*, Haifa, Israel, 22–25 June 1997. Berlin: Springer.

26. Yovine S. KRONOS: a verification tool for real-time systems. *Int J Software Tool Technol Trans* 1997; 1: 123–133.

27. Godskesen JC, Larsen KG and Skou A. *Automatic verification of real-time systems using epsilon*, 1995, https://www.brics.dk/RS/94/19/BRICS-RS-94-19.pdf

28. David A, Amnell T and Stigge M. Uppaal 4.0: small tutorial, http://www.it.uu.se/research/group/darts/uppaal/small_tutorial.pdf