

DEVICE CONTROL METHOD AND APPARATUS, SERVER, AND STORAGE MEDIUM

[0001] This application claims priority to Chinese Patent Application No. 202110956934.2, filed on August 19, 2021 and entitled "DEVICE CONTROL METHOD AND APPARATUS, SERVER, AND STORAGE MEDIUM", which is incorporated herein by reference in its entirety.

TECHNICAL FIELD

[0002] This application relates to the field of internet of things technologies, and in particular, to a device control method and apparatus, a server, and a storage medium.

BACKGROUND

[0003] With the rapid development of internet of things (internet of things, IoT) technologies, an IoT device may access an IoT platform to implement communication between people, machines, and things anytime, anywhere. In addition, an IoT application can control a corresponding IoT device through the IoT platform. In conventional technologies, developers of the IoT application develop, through the IoT platform, a digital twin model corresponding to a device. After the IoT device accesses the platform, the IoT device is bound to the corresponding digital twin model. In this way, the IoT application and the IoT device can perform control command delivery and data reporting by using the digital twin model, so that the IoT device is controlled.

[0004] In the foregoing technologies, because different data formats are set for devices of a same type produced by different manufactures, the application developers need to control the devices of different manufactures according to commands in different data formats using different digital twin models. As there is a strong coupling between the application and the device, making a process of controlling the IoT device very complex.

SUMMARY

[0005] To resolve a problem that a process of controlling a network device is complex, embodiments of this application provide a device control method and apparatus, a server, and a computer-readable storage medium. The technical solution is as follows.

[0006] A first aspect provides a device control method, performed by a server, and the method includes:

receiving a first control command of a control device, where the first control command carries a device identifier of a first network device and a first control entity, the first control
5 command instructs the first network device to perform an operation corresponding to the first control entity, the first control entity complies with a standard data format, and the standard data format is described by a standard model corresponding to an upper device type of the first network device;

determining, based on the device identifier of the first network device, a first digital
10 twin model corresponding to the first network device, where the first digital twin model describes a data format supported by the first network device, and the first digital twin model records a first correspondence between the standard data format and the data format supported by the first network device; and

converting, based on the first correspondence, the first control entity into the data
15 format supported by the first network device, and sending a converted first control entity to the first network device.

[0007] According to the device control method provided in this embodiment of this application, for network devices of any type, the network devices of the type include a plurality of different model numbers. Format conversion is performed, based on a correspondence that is recorded in a
20 digital twin model of a network device with any model number of the type and that is between a standard data format of the network device of the type and a data format supported by the network device with the model number, on a control entity of a control command sent by a control device. According to the method, the control device can control the network devices with different model numbers by using a control entity of a control command in a unified standard data format, and the
25 network device can identify the control entity of the control command sent by the control device without adapting the standard data format to data formats supported by the network devices with different model numbers. Decoupling between the network device and an internet of things application is implemented, thereby simplifying a process of controlling the network device.

[0008] In some embodiments, the method further includes:

receiving a second control command of the control device, where the second control
30 command carries a device identifier of a second network device and a second control entity, the second control command instructs the second network device to perform an operation corresponding to the second control entity, the second control entity complies with the standard data format, a model number of the first network device is different from a model number of the
35 second network device, and the upper device type of the first network device is the same as an

upper device type of the second network device; determining, based on the device identifier of the second network device, a second digital twin model corresponding to the second network device, where the second digital twin model describes a data format supported by the second network device, and the second digital twin model records a second correspondence between the standard data format and the data format supported by the second network device; and converting, based on the second correspondence, the second control entity into the data format supported by the second network device, and sending a converted second control entity to the second network device.

[0009] According to the method provided in this embodiment of this application, format conversion can be performed on the control entity of the second control command by using a correspondence between a standard data format of an upper device type and the data format supported by the second network device, and a converted control entity can be sent to the second network device, to control network devices with different model numbers of a same type by using a control entity of a control command in a unified standard data format.

[0010] In some embodiments, the method further includes:
receiving status data of the first network device and the device identifier of the first network device, where the status data is the data format supported by the first network device; and converting the status data into the standard data format based on the first correspondence, and sending converted status data to the control device. Format conversion is performed on the status data of the network device, so that the control device can identify status data of network devices with different model numbers in a unified standard data format.

[0011] In some embodiments, before the receiving a first control command of a control device, the method further includes:

receiving a device access request of the first network device, where the device access request carries a model number of the first network device and the device identifier of the first network device; determining, based on the model number of the first network device, a first digital twin model corresponding to the model number of the first network device; and establishing a third correspondence between the device identifier of the first network device and the first digital twin model. The third correspondence between the device identifier and the first digital twin model is established, so that the first network device is bound to the corresponding first digital twin model. Therefore, a format of the control entity of the first control command of the first network device can be converted.

[0012] In some embodiments, after the establishing a third correspondence between the device identifier of the first network device and the first digital twin model, the method further includes:
creating a first running instance of the first network device based on the first digital

twin model. The first running instance is created, so that the status data of the first network device can be stored. Therefore, a historical status of the first network device can be found.

[0013] In some embodiments, the first correspondence includes a correspondence between a digital twin metamodel of the first digital twin model and a standard metamodel of the standard model.

[0014] The converting, based on the first correspondence, the first control entity into the data format supported by the first network device includes:

determining one or more standard metamodels of the upper device type based on the correspondence between a digital twin metamodel of the first digital twin model and a standard metamodel of the standard model; and converting, based on the first digital twin model and the one or more standard metamodels of the upper device type, the first control entity into the data format supported by the first network device. A standard metamodel corresponding to the first network device is determined, so that format conversion can be performed on the first control entity by using a standard data format provided by the standard metamodel and the format supported by the first network device, to control the first network device.

[0015] In some embodiments, the first digital twin model includes one or more digital twin metamodels. The digital twin metamodel includes one or more of the following: a property metamodel, a command metamodel, an event metamodel, a data metamodel, a component metamodel, and a relationship metamodel. The first digital twin model is divided into a plurality of digital twin metamodels, so that the digital twin model can be divided into a plurality of submodels. Therefore, a plurality of features of a type of the first network device can be more flexibly described, so that the first network device can be associated with a corresponding standard model more easily.

[0016] A second aspect provides a device control apparatus. The apparatus includes a plurality of functional modules, configured to perform corresponding steps in the device control method provided in the first aspect.

[0017] According to the device control apparatus, a network device with any model number of any type performs, based on a correspondence that is recorded in a digital twin model of the network device with the model number and that is between a standard data format of the network device of the type and a data format supported by the network device with the model number, format conversion on a control entity of a control command sent by a control device, so that the control device can control network devices with different model numbers by using a control entity of a control command in a unified standard data format. In addition, the network device can identify the control entity of the control command sent by the control device without adapting the standard data format to data formats supported by the network devices with different model

numbers. Decoupling between the network device and an internet of things application is implemented, thereby simplifying a process of controlling the network device.

[0018] A third aspect provides a server. The server includes a communication bus, at least one network interface, a memory, and at least one processor. The communication bus is used to perform information transmission between the foregoing components. The at least one network interface is configured to communicate with another device or a communication network. The memory stores program code. The at least one processor implements any step of the device control method in the first aspect by reading and executing the program code stored in the memory.

[0019] According to the server, a network device with any model number of any type performs, based on a correspondence that is recorded in a digital twin model of the network device with the model number and that is between a standard data format of the network device of the type and a data format supported by the network device with the model number, format conversion on a control entity of a control command sent by a control device, so that the control device can control network devices with different model numbers by using a control entity of a control command in a unified standard data format. In addition, the network device can identify the control entity of the control command sent by the control device without adapting the standard data format to data formats supported by the network devices with different model numbers. Decoupling between the network device and an internet of things application is implemented, thereby simplifying a process of controlling the network device.

[0020] A fourth aspect provides a computer-readable storage medium. When program code in the computer-readable storage medium is executed by a processor of a server, the server is enabled to perform any step of the device control method in the first aspect.

[0021] According to the computer-readable storage medium, a network device with any model number of any type performs, based on a correspondence that is recorded in a digital twin model of the network device with the model number and that is between a standard data format of the network device of the type and a data format supported by the network device with the model number, format conversion on a control entity of a control command sent by a control device, so that the control device can control network devices with different model numbers by using a control entity of a control command in a unified standard data format. In addition, the network device can identify the control entity of the control command sent by the control device without adapting the standard data format to data formats supported by the network devices with different model numbers. Decoupling between the network device and an internet of things application is implemented, thereby simplifying a process of controlling the network device.

BRIEF DESCRIPTION OF DRAWINGS

[0022] FIG. 1 is a schematic diagram of an implementation environment of a device control method according to an embodiment of this application;

[0023] FIG. 2 is a schematic diagram of a structure of a server 200 according to an embodiment of this application;

[0024] FIG. 3 is a schematic diagram of a structure of an internet of things platform according to an embodiment of this application;

[0025] FIG. 4 is a flowchart of steps of a device control method according to an embodiment of this application;

[0026] FIG. 5 is a flowchart of steps of obtaining a standard model according to an embodiment of this application;

[0027] FIG. 6 shows a standard model creation page according to an embodiment of this application;

[0028] FIG. 7 is a flowchart of steps of obtaining a digital twin model according to an embodiment of this application;

[0029] FIG. 8 is a schematic diagram of a device control method according to an embodiment of this application;

[0030] FIG. 9 is a flowchart of steps of sending a control command according to an embodiment of this application; and

[0031] FIG. 10 is a schematic diagram of a structure of a device control apparatus according to an embodiment of this application.

DESCRIPTION OF EMBODIMENTS

[0032] To make objectives, technical solutions, and advantages of this application clearer, the following further describes implementations of this application in detail with reference to the accompanying drawings.

[0033] To facilitate understanding of technical processes of embodiments of this application, the following explains some key terms and nouns in embodiments of this application.

[0034] (1) Internet of things: The internet of things is an internet that connects things, and can implement interconnection between things and between people and things. The internet of things is a network that implements intelligent identification, positioning, tracking, monitoring, and management.

[0035] (2) Internet of things platform: The internet of things platform is an internet-based platform that connects objects and implements an internet of things function. The internet of things

platform has a plurality of functions such as communication between a network device and an internet of things application, device access, device management, data storage, and access of the internet of things application.

[0036] (3) Digital twin model: The digital twin model is a virtual model implemented based on a digital twin model language, and is used to describe a feature of the network device.

[0037] (4) Digital twin model language (digital twin model language, DTML): The digital twin model language is a computer program language based on a Javascript object notation for linked data (Javascript object notation for linked data, JSON-LD), and is used to describe the digital twin model.

[0038] The following describes an implementation environment of a device control method provided in embodiments of this application.

[0039] FIG. 1 is a schematic diagram of an implementation environment of a device control method according to an embodiment of this application. The implementation environment includes control devices 101, a server 102, and network devices 103. The control device 101 and the server 102 can be directly or indirectly connected via a wired network or a wireless network. The server 102 and the network device 103 can be directly or indirectly connected via a wired network or a wireless network. This is not limited in embodiments of this application.

[0040] The control device 101 may be a smartphone, a tablet computer, a notebook computer, a desktop computer, or the like, but is not limited thereto. An internet of things application is installed in the control device 101, and the internet of things application may be an industrial application, a logistics application, a life application, or the like. The control device 101 can send a control command to the network device 103 based on the internet of things application by using the server 102, control the network device to perform a corresponding operation, and receive, by using the server 102, status data sent by the network device 103.

[0041] The server 102 is a control server. The server 102 may be an independent physical server, or may be a server cluster or a distributed system including a plurality of physical servers, or may be a cloud server that provides basic cloud computing services such as a cloud service, a cloud database service, a cloud computing service, a cloud function service, cloud storage, a network service, cloud communication, a middleware service, a domain name service, a security service, a content delivery network (content delivery network, CDN), a big data service, and an artificial intelligence platform. An internet of things platform is deployed on the server 102. The server 102 can forward the control command and the status data between the control device 101 and the network device 103 through the internet of things platform.

[0042] The network devices 103 may be network devices of different types produced by different manufacturers, for example, a camera, an intelligent refrigerator, and an air conditioner.

This is not limited in the embodiments. The network device 103 can receive the control command of the control device 101 by using the server 102, perform the corresponding operation based on the control command, and send the status data to the control device 101 by using the server 102.

[0043] The following describes a structure of the server in embodiments of this application.

[0044] FIG. 2 is a schematic diagram of a structure of a server 200 according to an embodiment of this application. The server 200 shown in FIG. 2 is configured to perform an operation related to a device control method. As shown in FIG. 2, the server 200 includes at least one processor 201, a communication bus 202, a memory 203, and at least one network interface 204.

[0045] The processor 201 is, for example, a general-purpose central processing unit (central processing unit, CPU), a network processor (network processor, NP), a graphics processing unit (graphics processing unit, GPU), a neural-network processing unit (neural-network processing unit, NPU), a data processing unit (data processing unit, DPU), a microprocessor, or one or more integrated circuits configured to implement solutions of this application. For example, the processor 201 includes an application-specific integrated circuit (application-specific integrated circuit, ASIC), a programmable logic device (programmable logic device, PLD), or a combination thereof. The PLD is, for example, a complex programmable logic device (complex programmable logic device, CPLD), a field programmable gate array (field programmable gate array, FPGA), a generic array logic (generic array logic, GAL), or any combination thereof.

[0046] The communication bus 202 is used to perform information transmission between the foregoing components. The communication bus 202 may be optionally classified into an address bus, a data bus, a control bus, and the like. For ease of representation, only one thick line is used for representing the bus in FIG. 2, but this does not mean that there is only one bus or only one type of bus.

[0047] The memory 203 is, for example, a read-only memory (read-only memory, ROM) or another type of static storage device that can store static information and instructions, a random access memory (random access memory, RAM) or another type of dynamic storage device that can store information and instructions, an electrically erasable programmable read-only memory (electrically erasable programmable read-only memory, EEPROM), a compact disc read-only memory (compact disc read-only memory, CD-ROM) or another optical disk storage, an optical disk storage (including a compressed optical disc, a laser disc, an optical disc, a digital versatile disc, a Blu-ray disc, or the like), a magnetic disk storage medium or another magnetic storage device, or any other medium that can be configured to carry or store desired program code in the form of instructions or a data structure and that can be accessed by a computer, but is not limited thereto. For example, the memory 203 exists independently, and is connected to the processor 201 through the communication bus 202. Alternatively, the memory 203 and the processor 201 are

optionally integrated together.

[0048] The network interface 204 uses any apparatus, for example, a transceiver, to communicate with another device or a communication network. The network interface 204 includes a wired network interface, and optionally further includes a wireless network interface.

5 The wired network interface is, for example, an Ethernet interface. The Ethernet interface is, for example, an optical interface, an electrical interface, or a combination thereof. The wireless network interface is, for example, a wireless local area network (wireless local area network, WLAN) interface, a cellular network interface, or a combination thereof.

[0049] In some embodiments, the processor 201 includes one or more CPUs such as a CPU 0
10 and a CPU 1 shown in FIG. 2.

[0050] In some embodiments, the server 200 optionally includes a plurality of processors such as a processor 201 and a processor 205 shown in FIG. 2. Each of the processors is, for example, a single-core processor, or a multi-core processor. Optionally, the processor herein refers to one or more devices, circuits, and/or processing cores configured to process data (for example, computer
15 program instructions).

[0051] In some embodiments, the server 200 further includes an output device and an input device. The output device optionally displays information in a plurality of manners. For example, the output device may optionally be a liquid crystal display (liquid crystal display, LCD), a light emitting diode (light emitting diode, LED) display device, a cathode ray tube (cathode ray tube, CRT) display device, or a projector (projector). The input device communicates with the processor
20 201, and optionally receives input of a user in a plurality of manners. For example, the input device may optionally be a mouse, a keyboard, a touchscreen device, or a sensing device.

[0052] Optionally, the processor 201 implements the method in the following embodiments by reading program code 210 stored in the memory 203, or the processor 201 implements the method
25 in the following embodiments by using program code stored internally. When the processor 201 implements the method in the following embodiments by reading the program code 210 stored in the memory 203, the memory 203 stores program code for implementing the device control method provided in embodiments of this application.

[0053] For more details of implementing the foregoing functions by the processor 201, refer to descriptions in the following method embodiments. Details are not described herein.
30

[0054] An embodiment of this application further provides a computer-readable storage medium including program code, for example, a memory 203 including the program code. The program code may be executed by a processor 201 of a server 200 to complete a device control method. Optionally, the computer-readable storage medium may be a read-only memory (read-only memory, ROM), a random access memory (random access memory, RAM), a compact disc
35

read-only memory (compact disc read-only memory, CD-ROM), a magnetic tape, a floppy disk, an optical data storage device, or the like.

[0055] Based on the foregoing implementation environment, the following describes the internet of things platform deployed on the server.

5 **[0056]** FIG. 3 is a schematic diagram of a structure of an internet of things platform according to an embodiment of this application. It should be understood that FIG. 3 merely shows an example of a schematic structural diagram of the internet of things platform. Division of functions of the internet of things platform is not limited in this application. As shown in FIG. 3, the internet of things platform provides three types of services, including a device access management service, a
10 model engine service, and a standard model service. The following separately describes the three types of services.

[0057] The device access management service is used to associate an internet of things application, a network device, and a corresponding digital twin model, and provide functions of device access and application access, to implement interworking between the internet of things
15 application and the network device. The device access management service is further used to send a received control command and received status data to the model engine service, receive status data and a control entity of the control command and whose formats are converted and that are sent by the model engine service, and forward the control entity and the status data to a corresponding network device or internet of things application.

20 **[0058]** The model engine service is a functional unit implemented by using a DTML, and the functional unit is configured to: synchronize a standard model in the standard model service, and generate a standard application programming interface (application programming interface, API) based on the standard model. The model engine service is further used to store and parse a digital twin model, perform data format conversion on the control entity of the control command and the
25 status data that are sent by the device access management service, and send the converted control entity of the control command and the converted status data to the device access management service. For a detailed process, refer to an embodiment corresponding to FIG. 4.

[0059] The standard model service is a functional unit implemented by using the DTML, and the functional unit is configured to: create and store a standard model.

30 **[0060]** Based on descriptions of a system architecture to which embodiments of this application are applied and a structure of the internet of things platform, the following describes a device control method provided in embodiments of this application with reference to FIG. 4. FIG. 4 is a flowchart of steps of a device control method according to an embodiment of this application. The method is performed by a server. As shown in FIG. 4, the device control method includes the
35 following steps 401 to 410.

[0061] 401: The server obtains a standard model, where the standard model describes a standard data format of an upper device type of a first network device.

[0062] The upper device type of the first network device refers to a device type to which the first network device belongs. The first network device may be a network device with any model number of the device type. For example, the upper device type is a camera, and the first network device is a camera with a first model number.

[0063] If a standard data format needs to be provided for any upper device type to simplify a process of controlling a network device of the upper device type, a platform administrator creates a standard model of the network device of the upper device type on the server. The standard model may be a standard metamodel group. In other words, the standard model may include one or more standard metamodels, and each standard metamodel is used to describe one feature of the upper device type. The standard metamodel includes one or more of a standard property metamodel (property), a standard command metamodel (command), a standard event metamodel (event), a standard data metamodel (dataschema), a standard component metamodel (component), and a standard relationship metamodel (relationship). The standard model is divided into a plurality of standard metamodels, and the standard model can be divided into a plurality of submodels. Therefore, a plurality of features of the upper device type can be more flexibly described, so that a digital twin model can be associated with the standard model more easily subsequently.

[0064] In some embodiments, a process of obtaining the standard model by the server is shown in FIG. 5. A standard model service of an internet of things platform provides a standard model creation page shown in FIG. 6. The platform administrator can compile model code on the standard model creation page, to create the standard model. The server is in response to a submission operation performed on the model code of the standard model on the standard model creation page, and the standard model service stores the model code, that is, stores the standard model. It may be understood that the standard model may include one or more standard metamodels, standard models corresponding to different upper device types may include different standard metamodels, and the platform administrator can create different standard models based on features of the different upper device types. This is not limited in this embodiment of this application.

[0065] It should be noted that, when creating a standard model of a network device of any upper device type, the platform administrator may first learn of a function and a property of the network device of the upper device type by querying device documents provided by manufacturers of a plurality of network devices of the upper device type, and then create the standard model, so that the created standard model can accurately and comprehensively describe a feature of the network device of the upper device type.

[0066] For example, an example in which the upper device type is a camera is used for

description. A camera standard model includes two standard metamodels: a standard command metamodel and a standard property metamodel. The platform administrator can compile model code on the standard model creation page, and in response to a submission operation performed on the model code of the camera standard model, the standard model service stores the camera standard model. The model code of the camera standard model is as follows:

```

5      "rotateCommand":{    //metamodel identifier of the standard command metamodel
        "displayName":"camera rotation",
        "input":{    //input parameter
          "@type": "object",
10      "displayName":"parameters: horizontal rotation angle, vertical rotation angle,
and speed",
          "fields":{
            "horizontalAngle": {    //identifier of the horizontal rotation angle
parameter
15      "@type":"integer"
          },
            "verticalAngle": {    //identifier of the vertical rotation angle parameter
              "@type":"integer"
            },
20      "speed": {    //identifier of the speed parameter
              "@type":"integer"
            }
          }
        },
25      "output": {    //output parameter
        "@type": "object",
        "fields": {
          "SUCCESS": {
            "@type": "integer"
30      }
          }
        }
      },
35      "status":{omitted}    //metamodel identifier of the standard property metamodel

```

[0067] A standard metamodel identified by rotateCommand is a standard command metamodel, and is configured to provide a standard data format of the rotate command of the camera. The rotate command includes three input parameters: the horizontal rotation angle, the vertical rotation angle, and the speed. Standard data formats of the three parameters are respectively formats described in code segments corresponding to horizontalAngle, verticalAngle, and speed. A standard metamodel identified by status is a standard property metamodel, and is configured to provide a standard data format of status data of the camera.

[0068] In some embodiments, the server can generate a standard application programming interface based on the standard model. Step 401 further includes: The standard model service sends a stored standard model to a model engine service, and the model engine service receives and stores the standard model to complete synchronization of the standard model, parses the standard model, and generates the standard application programming interface for each standard metamodel of the standard model. The standard application programming interface can be used for development of an internet of things application, so that the internet of things application can send or receive a control entity of a control command and status data that are in the standard data format.

[0069] For example, an example in which a standard application programming interface is generated for the camera standard model is used for description. The model engine service synchronizes the camera standard model from the standard model service, parses the camera standard model, and generates the standard application programming interface including a rotate command interface and a status property interface for two standard metamodels. The standard application programming interface of the two standard metamodels is shown as follows:

Https://{host}/device/(deviceid)/rotateCommand //rotate command interface

Https://{host}/device/(deviceid)/status //status property interface

[0070] 402: The server obtains a first digital twin model, and associates the first digital twin model with a corresponding standard model, where the first digital twin model describes a data format supported by the first network device.

[0071] For any upper device type, if a network device with any model number of the upper device type (for example, a network device with a first model number of the upper device type) needs to be controlled using the internet of things application, an application developer creates, on the internet of things platform, a digital twin model corresponding to the network device with the first model number, and develops the internet of things application.

[0072] The first digital twin model is a first digital twin metamodel group. The first digital twin model includes one or more digital twin metamodels, and each digital twin metamodel is used to describe one feature of the first network device. The digital twin metamodel includes one or more of the following: a property metamodel, a command metamodel, an event metamodel, a data

metamodel, a component metamodel, and a relationship metamodel.

[0073] In some embodiments, the application developer first registers with the internet of things platform, the internet of things platform generates a user identifier for the application developer, and then the application developer creates the first digital twin model corresponding to the first network device. A process in which the server obtains the first digital twin model is shown in FIG. 7. The internet of things platform provides a digital twin model creation page. The application developer can compile, on the digital twin model creation page, model code of each digital twin metamodel in the first digital twin model based on the data format supported by the first network device, find a standard model of the upper device type of the first network device by using the standard model service of the internet of things platform, and associate the created first digital twin model with the corresponding standard model of the upper device type, in other words, associate the created first digital twin metamodel group with a corresponding standard metamodel group. After the model code is compiled, the application developer can submit the model code on the digital twin model creation page.

[0074] For example, the foregoing process of associating the first digital twin model with the corresponding standard model is described. When compiling the model code of each digital twin metamodel in the first digital twin model, the application developer adds an element of "@type" to the code of each digital twin metamodel in the created first digital twin model, and adds, after the element of "@type", a metamodel identifier of a standard metamodel in a standard model corresponding to each digital twin metamodel. Therefore, each digital twin metamodel in the created first digital twin model can be associated with the corresponding standard metamodel in the corresponding standard model by using the element of "@type", the element of "@type" is added in code corresponding to each parameter of each digital twin metamodel, and an identifier of a corresponding parameter in a standard metamodel corresponding to each parameter in each digital twin metamodel is added after the element of "@type", to associate each parameter in each digital twin metamodel with the corresponding parameter in the corresponding standard metamodel by using the element of "@type". Each digital twin metamodel of the first digital twin model is associated with each standard metamodel of the corresponding standard model, and each parameter of each digital twin metamodel is associated with each parameter of each corresponding standard metamodel, so that the first digital twin model is associated with the corresponding standard model.

[0075] The first digital twin model provides the data format supported by the first network device, and the standard model provides the standard data format. Therefore, the first digital twin model records, by using the element of "@type", a first correspondence between the standard data format and the data format supported by the first network device, and the first correspondence

includes a correspondence between a digital twin metamodel of the first digital twin model and a standard metamodel of the standard model, so that a model engine service can implement, by parsing an element of "@type" of each digital twin metamodel in the first digital twin model, conversion between the data format supported by the first network device and the standard data format.

[0076] In some embodiments, in response to a submission operation performed on the model code on the digital twin model creation page, a device access service sends a model creation request to the model engine service, where the model creation request carries the model code of the first digital twin model, the model creation request indicates to store the created first digital twin model, and the model engine service stores the first digital twin model, obtains a model identifier of the first digital twin model by parsing an id field in the first digital twin model, and identifies, by parsing the element of "@type" of each digital twin metamodel in the first digital twin model, a standard metamodel associated with the digital twin metamodel.

[0077] It should be noted that the model identifier of the first digital twin model is obtained based on a model number of the first network device. In some embodiments, before creating the first digital twin model of the first network device, the application developer obtains the model number of the first network device from a manufacturer of the first network device. In some cases, the model number of the first network device may be a character string. Optionally, when creating the first digital twin model, the application developer compiles the id field in the model code by using a target field in the model number of the first network device, to create the model identifier of the first digital twin model. Alternatively, when submitting the model code on the digital twin model creation page, the application developer simultaneously submits the model number of the first network device, and in response to a submission operation performed on the model code and the model of the first network device on the digital twin model creation page, the device access service, uses the target field in the model number of the first network device as the model identifier of the first digital twin model, and generates the id field in the model code based on the model identifier.

[0078] For example, creating a digital twin model of a camera with an A01 model number is used as an example for description. It should be understood that A01 is merely an abbreviation for a model number of the camera, and the model number of the camera is a character string. The application developer obtains the model number character string of the camera with the A01 model number, uses a target field in the character string as a model identifier, and compiles model code of the digital twin model of the camera with the A01 model number on the digital twin model creation page based on a data format supported by the camera with the A01 model number. The digital twin model includes two digital twin metamodels: a command metamodel commandA01

and a property metamodel PropertyA01. Through the element of "@type", commandA01 is associated with a standard metamodel rotateCommand, and PropertyA01 is associated with a standard metamodel status. The model code of the camera with the A01 model number is as follows:

```

    {
5      "id":"dtml:dahua:Camera;1", //model identifier
      "properties":{    //property metamodel
        "propertyA01":{
          "@type":"status"    //associated with the standard metamodel identified
by status
10      }
        },
      "command":{    // command metamodel
        "commandA01":{
          "@type":"rotateCommand", //associated with the standard metamodel
15 identified by rotateCommand
          "displayName":"camera rotation",
          "input":{
            "@type": "object",
            "displayName":"parameters: horizontal rotation angle, vertical
20 rotation angle, and speed",
            "fields":{
              "ha": {    //horizontal rotation angle parameter
                "@type":"integer, horizontalAngle " //associated with
horizontalAngle
25              },
              "va": {    //vertical rotation angle parameter
                "@type":"integer, verticalAngle " // associated with
verticalAngle
30              },
              "speed": {    //speed parameter
                "@type":"integer, speed " // associated with speed
              }
            }
          }
        }
      }
35    }

```

}
}

[0079] A digital twin metamodel identified by commandA01 is a command metamodel, and is configured to provide a data format supported by a rotate command of the camera with the A01 model number. The rotate command includes three input parameters: the horizontal rotation angle, the vertical rotation angle, and the speed. For the three parameters, the data formats supported by the camera with the A01 model number are respectively formats described by code segments corresponding to ha, va, and speed. A digital twin metamodel identified by propertyA01 is a property metamodel, and is configured to provide a data format supported by status data of the camera with the A01 model number.

[0080] It should be noted that data formats supported by network devices with different model numbers of a same upper device type are different. Therefore, different digital twin models need to be created based on the data formats supported by the network devices with different model numbers, and the digital twin models of the network devices with different model numbers are associated with a standard model of the upper device type. For example, an example in which the upper device type is a camera is used for description. As shown in FIG. 8, for a camera with an A01 model number, a camera with a B01 model number, and a camera with a C01 model number, digital twin models corresponding to the cameras with the three model numbers may be separately created, and the digital twin models corresponding to the cameras with the three model numbers are separately associated with a camera standard model.

[0081] Digital twin models of network devices with a plurality of model numbers are associated with a standard model of the upper device type of the network device. When developing an internet of things application for the network device, the application developer can enable the network device to identify a control entity of a control command sent by a control device without adapting, one by one, data formats supported by the network devices with different model numbers, so that a process of controlling the network device is simplified.

[0082] 403: The first network device sends a device access request to the server, where the device access request carries the model number of the first network device and a device identifier of the first network device.

[0083] The target field of the model number of the first network device is the model identifier of the first digital twin model corresponding to the first network device.

[0084] In some embodiments, after creating the first digital twin model of the first network device, the application developer notifies the manufacturer of the first network device of address information of the server. The manufacturer can preset the address information of the server in a configuration file of the first network device, and the model number and the device identifier of

the first network device are further preset in the first network device. When the first network device is started for the first time and accesses a network, the first network device can send the device access request to the server based on the address information of the server, where the device access request carries the model number and the device identifier of the first network device.

5 **[0085]** 404: The server receives the device access request of the first network device, and associates the first network device with the corresponding first digital twin model.

[0086] In some embodiments, the step 404 includes the following two processes:

[0087] Process 1: Associate the first network device with the corresponding first digital twin model.

10 **[0088]** In some embodiments, in response to the device access request of the first network device, a device access management service obtains the target field in the model number of the first network device carried in the device access request; determines a model identifier that is the same as the target field, where a digital twin model corresponding to the model identifier is the first digital twin model corresponding to the first network device; correspondingly stores the
15 device identifier of the first network device and the model identifier, to establish a third correspondence between the device identifier of the first network device and the first digital twin model; and sends the model identifier to the model engine service.

[0089] Process 2: Create a running instance of the first network device.

[0090] In some embodiments, the model engine service creates the running instance of the first
20 network device based on the first digital twin model corresponding to the model identifier, and generates an instance identifier corresponding to the running instance. The running instance is used to store status data of the first network device. The model engine service sends the instance identifier of the first network device to the device access management service, and the device access management service stores information about a correspondence between the instance
25 identifier, the model identifier, and the device identifier of the first network device, to associate the first network device with the corresponding first digital twin model and the corresponding running instance.

[0091] 405: The control device sends a first control command of the first network device to the server, where the first control command carries the device identifier of the first network device
30 and a first control entity, the first control command instructs the first network device to perform an operation corresponding to the first control entity, and the first control entity is a standard data format.

[0092] In some embodiments, the control device sends the first control command using the internet of things application. The internet of things application is an application program
35 developed by the application developer based on the standard application programming interface,

and the first control command carries the device identifier of the first network device and the first control entity in the standard format. Optionally, the internet of things application provides a control page. A user of the internet of things application can subscribe to the first network device on the control page, and send the first control command of the first network device to the server.

5 **[0093]** For example, as shown in FIG. 9, an example in which the control device sends a rotate command for a camera with an A01 model number to the server is used for description. The control device displays the control page of the internet of things application, displays, on the control page, a plurality of network devices that have accessed the internet of things platform, displays a parameter input page of the rotate command of the camera with the A01 model number in response
10 to a click operation on the camera with the A01 model number, generates a control entity of the rotate command for the server based on a command parameter in the parameter input page in response to a submission operation on the parameter input page, and sends a rotateCommand to the server, where the rotateCommand carries a device identifier of the camera with the A01 model number and the control entity of the rotate command.

15 **[0094]** In some embodiments, the device access management service receives the first control command of the control device, where the first control command instructs the first network device to perform the operation corresponding to the first control entity. For example, the device access management service receives a rotate command of the control device. The rotate command carries a device identifier and a control entity, and the control entity in the rotate command indicates a
20 camera corresponding to the device identifier to rotate by a target angle at a target speed.

[0095] 406: The server receives the first control command of the control device, and determines, based on the device identifier of the first network device, the first digital twin model corresponding to the first network device, where the first digital twin model records the first correspondence between the standard data format and the data format supported by the first
25 network device.

[0096] The first digital twin model records, by using the element of "@type", the first correspondence between the standard data format and the data format supported by the first network device.

[0097] In some embodiments, in response to receiving the first control command of the control
30 device, the device access management service determines, based on a correspondence between the device identifier and the model identifier stored in the foregoing step 404, a digital twin model indicated by the model identifier corresponding to the device identifier of the first network device as the first digital twin model corresponding to the first network device, and forwards the first control command and the model identifier of the first digital twin model to the model engine
35 service.

[0098] 407: The server converts, based on the first correspondence, the first control entity into the data format supported by the first network device, and sends a converted first control entity to the first network device.

[0099] The first correspondence includes a correspondence between a digital twin metamodel of the first digital twin model and a standard metamodel of a standard model. The standard model may be a standard metamodel group, and the standard model may include one or more standard metamodels. Correspondingly, the server determines one or more standard metamodels of the upper device type of the first network device based on the correspondence between a digital twin metamodel of the first digital twin model and a standard metamodel of a standard model, converts, based on the first digital twin model and the one or more standard metamodels of the upper device type, the first control entity into the data format supported by the first network device, and sends a converted first control entity to the first network device.

[00100] In some embodiments, the model engine service parses the first digital twin model, determines a standard command metamodel corresponding to a metamodel identifier corresponding to the element of "@type" of a command metamodel in the first digital twin model, and determines the standard command metamodel as a standard metamodel corresponding to the upper device type of the first network device. The model engine service converts, based on a data format described by the command metamodel in the first digital twin model, a standard data format described by the standard command metamodel, and a correspondence between each parameter in the command metamodel and each parameter in the standard command metamodel, the first control entity from the standard data format to the data format supported by the first network device, and sends a converted first control instruction to the device access management service, where the converted first control instruction carries a converted first control entity and the device identifier of the first network device. The device access management service sends the converted first control entity to the first network device based on the device identifier, or sends the converted first control instruction to the first network device.

[00101] For example, as shown in FIG. 9, an example in which a rotate command is sent to a camera with an A01 model number is used for description. The model engine service receives a rotateCommand, performs format conversion on three parameters in a control entity of the rotateCommand based on a correspondence between each parameter in a command metamodel of a digital twin model corresponding to the camera with the A01 model number and each parameter in a corresponding standard command metamodel, converts a format of horizontalAngle into a format of ha, converts a format of verticalAngle into a format of va, and converts a format of speed into a format of speed, to obtain a control entity of a commandA01, to be specific, obtain a control entity of a data format supported by the camera with the A01 model number. The control entity of

the commandA01 and a device identifier of the camera with the A01 model number are sent to the device access management service, where the device access management service sends the control entity of the commandA01 to the camera with the A01 model number.

[00102] 408: The first network device sends the status data and the device identifier of the first network device to the control device by using the server, where the status data is the data format supported by the first network device.

[00103] The status data indicates a status of the first network device. For example, if the first network device is a camera, the status data indicates a rotation angle, a remaining power, a temperature, and the like of the camera.

[00104] In some embodiments, the first network device sends the status data of the first network device to the control device by using the server, where the status data is the data format supported by the first network device. For example, the first network device is a camera with any model number. As shown in FIG. 8, data formats of status data sent by cameras with different model numbers are different. Status data sent by the camera with the A01 model number is propertyA01, status data sent by the camera with the B01 model number is propertyB01, and status data sent by the camera with the C01 model number is propertyC01.

[00105] 409: The server receives the device identifier and the status data of the first network device, and determines, based on the device identifier of the first network device, the first digital twin model corresponding to the first network device.

[00106] In some embodiments, in response to receiving the device identifier and the status data of the first network device, the device access management service determines, based on a method similar to that in step 406, the first digital twin model corresponding to the first network device. Details are not described herein again.

[00107] 410: The server converts the status data into the standard data format based on the first correspondence, and sends converted status data to the control device.

[00108] In some embodiments, the model engine service converts, based on a method similar to that in step 407, the status data from the data format supported by the first network device into the standard data format, stores converted status data, and sends the converted status data to the device access management service. The device access management service sends the converted status data to the control device.

[00109] It should be noted that the server can be configured to perform format conversion on control commands of a plurality of network devices, to control the plurality of network devices. For network devices with a plurality of different model numbers of a same upper device type, the network devices with the plurality of different model numbers correspond to different digital twin models.

[00110] In some embodiments, the server can perform format conversion on a second control command based on a correspondence recorded in a second digital twin model of a second network device, to control the second network device. A model number of the second network device is different from the model number of the first network device, and an upper device type of the second network device is the same as the upper device type of the first network device. Correspondingly, this embodiment of this application further includes: The server obtains the second digital twin model, and associates the second digital twin model with a corresponding standard model. A standard model corresponding to the second network device is the same as the standard model corresponding to the first network device, and the second digital twin model describes a data format supported by the second network device. The server receives the second control command of the control device, where the second control command carries a device identifier of the second network device and a second control entity, the second control command instructs the second network device to perform an operation corresponding to the second control entity, and the second control entity complies with a standard data format. The server determines, based on the device identifier of the second network device, a second digital twin model corresponding to the second network device, where the second digital twin model records a second correspondence between the standard data format and the data format supported by the second network device. The server converts, based on the second correspondence, the second control entity into the data format supported by the second network device, and sends a converted second control entity to the second network device. A specific implementation of the foregoing process is similar to step 402 to step 407. Details are not described herein again.

[00111] It should be noted that the foregoing embodiments are described by using an example in which a control command and status data are exchanged between a control device and a network device. In some embodiments, the method provided in this application may only include that the control device sends the control command to the network device by using a server, or only include that the network device sends the status data to the control device by using the server.

[00112] According to the device control method provided in this embodiment of this application, for network devices of any type, the network devices of the type include a plurality of different model numbers. Format conversion is performed, based on a correspondence that is recorded in a digital twin model of a network device with any model number of the type and that is between a standard data format of the network device of the type and a data format supported by the network device with the model number, on a control entity of a control command sent by a control device. According to the method, the control device can control the network devices with different model numbers by using a control entity of a control command in a unified standard data format, and the network device can identify the control entity of the control command sent by the control device

without adapting the standard data format to data formats supported by the network devices with different model numbers. Decoupling between the network device and an internet of things application is implemented, thereby simplifying a process of controlling the network device.

[00113] In the foregoing embodiments, when a digital twin model is created, "@type" indicates a correspondence between a digital twin metamodel in the digital twin model and a standard metamodel in a standard model. It may be understood that the foregoing correspondence may alternatively be indicated by using another preset identifier. This is not limited in this application.

[00114] FIG. 10 is a schematic diagram of a structure of a device control apparatus according to an embodiment of this application. The device control apparatus is configured to perform steps performed when the foregoing device control method is performed. Refer to FIG. 10. The device control apparatus includes a receiving module 1001, a determining module 1002, and a conversion module 1003.

[00115] The receiving module 1001 is configured to receive a first control command of a control device, where the first control command carries a device identifier of a first network device and a first control entity, the first control command instructs the first network device to perform an operation corresponding to the first control entity, the first control entity complies with a standard data format, and the standard data format is described by a standard model corresponding to an upper device type of the first network device.

[00116] The determining module 1002 is configured to determine, based on the device identifier of the first network device, a first digital twin model corresponding to the first network device, where the first digital twin model describes a data format supported by the first network device, and the first digital twin model records a first correspondence between the standard data format and the data format supported by the first network device.

[00117] The conversion module 1003 is configured to convert, based on the first correspondence, the first control entity into the data format supported by the first network device, and send a converted first control entity to the first network device.

[00118] In some embodiments, the receiving module 1001 is configured to receive a second control command of the control device, where the second control command carries a device identifier of a second network device and a second control entity, the second control command instructs the second network device to perform an operation corresponding to the second control entity, the second control entity complies with the standard data format, a model number of the first network device is different from a model number of the second network device, and the upper device type of the first network device is the same as an upper device type of the second network device.

[00119] The determining module 1002 is configured to determine, based on the device identifier

of the second network device, a second digital twin model corresponding to the second network device, where the second digital twin model describes a data format supported by the second network device, and the second digital twin model records a second correspondence between the standard data format and the data format supported by the second network device.

5 **[00120]** The conversion module 1003 is configured to: convert, based on the second correspondence, the second control entity into the data format supported by the second network device, and send a converted second control entity to the second network device.

10 **[00121]** In some embodiments, the receiving module 1001 is configured to receive status data of the first network device and the device identifier of the first network device, where the status data is the data format supported by the first network device.

[00122] The conversion module 1003 is configured to: convert the status data into the standard data format based on the first correspondence, and send converted status data to the control device.

15 **[00123]** In some embodiments, the receiving module 1001 is configured to receive a device access request of the first network device, where the device access request carries the model number of the first network device and the device identifier of the first network device.

[00124] The determining module 1002 is configured to determine, based on the model number of the first network device, a first digital twin model corresponding to the model number of the first network device.

20 **[00125]** The apparatus further includes an establishing module, configured to establish a third correspondence between the device identifier of the first network device and the first digital twin model.

[00126] In some embodiments, the first correspondence includes a correspondence between a digital twin metamodel of the first digital twin model and a standard metamodel of the standard model.

25 **[00127]** The conversion module 1003 is configured to: determine one or more standard metamodels of the upper device type based on the correspondence between a digital twin metamodel of the first digital twin model and a standard metamodel of the standard model; and convert, based on the first digital twin model and the one or more standard metamodels of the upper device type, the first control entity into the data format supported by the first network device.

30 **[00128]** In some embodiments, the first digital twin model includes one or more digital twin metamodels. The digital twin metamodel includes one or more of the following: a property metamodel, a command metamodel, an event metamodel, a data metamodel, a component metamodel, and a relationship metamodel.

35 **[00129]** It should be noted that, when the device control apparatus provided in the embodiment corresponding to FIG. 10 performs device control, division of the foregoing functional modules is

merely used as an example for description. During actual application, the foregoing functions may be allocated to different functional modules for implementation based on a requirement. In other words, an internal structure of the apparatus is divided into different functional modules to implement all or some of the functions described above. In addition, the device control apparatus
5 provided in the foregoing embodiments and the device control method embodiments belong to a same concept. For a specific implementation process thereof, refer to the method embodiments. Details are not described herein again.

[00130] In this application, the term "at least one" means one or more, and the term "a plurality of" means two or more. For example, a plurality of standard metamodels means two or more
10 standard metamodels.

[00131] It should be further understood that the term "if" may be interpreted as "when" ("when" or "upon"), "in response to determining", or "in response to detecting". Similarly, based on the context, the phrase "if it is determined ..." or "if [the stated condition or event] is detected" may be interpreted as "when ... is determined", "in response to determining ..." "when [the stated condition
15 or event] is detected", or "in response to detecting [the stated condition or event]".

[00132] The foregoing descriptions are merely specific implementations of this application, but are not intended to limit the protection scope of this application. Various equivalent modifications or replacements readily figured out by any person skilled in the art within the technical scope disclosed in this application shall fall within the protection scope of this application. Therefore,
20 the protection scope of this application shall be subject to the protection scope of the claims.

[00133] A person of ordinary skill in the art may understand that all or some of the steps of the foregoing embodiments may be implemented by hardware or a program instructing related hardware. The program may be stored in a computer-readable storage medium. The storage medium may be a read-only memory, a magnetic disk, an optical disc, or the like.

[00134] The foregoing descriptions are merely optional embodiments of this application, but are not intended to limit this application. Any modification, equivalent replacement, or improvement made without departing from the spirit and principle of this application should fall within the protection scope of this application.

CLAIMS

What is claimed is:

1. A device control method, performed by a server, wherein the method comprises:

receiving a first control command of a control device, wherein the first control command
5 carries a device identifier of a first network device and a first control entity, the first control command instructs the first network device to perform an operation corresponding to the first control entity, the first control entity complies with a standard data format, and the standard data format is described by a standard model corresponding to an upper device type of the first network device;

10 determining, based on the device identifier of the first network device, a first digital twin model corresponding to the first network device, wherein the first digital twin model describes a data format supported by the first network device, and the first digital twin model records a first correspondence between the standard data format and the data format supported by the first network device; and

15 converting, based on the first correspondence, the first control entity into the data format supported by the first network device, and sending a converted first control entity to the first network device.

2. The method according to claim 1, wherein the method further comprises:

receiving a second control command of the control device, wherein the second control
20 command carries a device identifier of a second network device and a second control entity, the second control command instructs the second network device to perform an operation corresponding to the second control entity, the second control entity complies with the standard data format, a model number of the first network device is different from a model number of the second network device, and the upper device type of the first network device is the same as an
25 upper device type of the second network device;

determining, based on the device identifier of the second network device, a second digital twin model corresponding to the second network device, wherein the second digital twin model describes a data format supported by the second network device, and the second digital twin model records a second correspondence between the standard data format and the data format supported
30 by the second network device; and

converting, based on the second correspondence, the second control entity into the data format supported by the second network device, and sending a converted second control entity to the second network device.

3. The method according to claim 1, wherein the method further comprises:

receiving status data of the first network device and the device identifier of the first network device, wherein the status data is the data format supported by the first network device; and
converting the status data into the standard data format based on the first correspondence, and sending converted status data to the control device.

5 4. The method according to claim 1, wherein before the receiving a first control command of a control device, the method further comprises:

receiving a device access request of the first network device, wherein the device access request carries a model number of the first network device and the device identifier of the first network device;

10 determining, based on the model number of the first network device, a first digital twin model corresponding to the model number of the first network device; and

establishing a third correspondence between the device identifier of the first network device and the first digital twin model.

15 5. The method according to claim 4, wherein the first correspondence comprises a correspondence between a digital twin metamodel of the first digital twin model and a standard metamodel of the standard model; and

the converting, based on the first correspondence, the first control entity into the data format supported by the first network device comprises:

20 determining one or more standard metamodels of the upper device type based on the correspondence between a digital twin metamodel of the first digital twin model and a standard metamodel of the standard model; and

converting, based on the first digital twin model and the one or more standard metamodels of the upper device type, the first control entity into the data format supported by the first network device.

25 6. The method according to any one of claims 1 to 5, wherein the first digital twin model comprises one or more digital twin metamodels, and the digital twin metamodel comprises one or more of the following: a property metamodel, a command metamodel, an event metamodel, a data metamodel, a component metamodel, and a relationship metamodel.

7. A device control apparatus, wherein the apparatus comprises:

30 a receiving module, configured to receive a first control command of a control device, wherein the first control command carries a device identifier of a first network device and a first control entity, the first control command instructs the first network device to perform an operation corresponding to the first control entity, the first control entity complies with a standard data format, and the standard data format is described by a standard model corresponding to an upper device
35 type of the first network device;

a determining module, configured to determine, based on the device identifier of the first network device, a first digital twin model corresponding to the first network device, wherein the first digital twin model describes a data format supported by the first network device, and the first digital twin model records a first correspondence between the standard data format and the data format supported by the first network device; and

a conversion module, configured to: convert, based on the first correspondence, the first control entity into the data format supported by the first network device, and send a converted first control entity to the first network device.

8. The apparatus according to claim 7, wherein the receiving module is configured to receive a second control command of the control device, wherein the second control command carries a device identifier of a second network device and a second control entity, the second control command instructs the second network device to perform an operation corresponding to the second control entity, the second control entity complies with the standard data format, a model number of the first network device is different from a model number of the second network device, and the upper device type of the first network device is the same as an upper device type of the second network device;

the determining module is configured to determine, based on the device identifier of the second network device, a second digital twin model corresponding to the second network device, wherein the second digital twin model describes a data format supported by the second network device, and the second digital twin model records a second correspondence between the standard data format and the data format supported by the second network device; and

the conversion module is configured to: convert, based on the second correspondence, the second control entity into the data format supported by the second network device, and send a converted second control entity to the second network device.

9. The apparatus according to claim 7, wherein the receiving module is configured to receive status data of the first network device and the device identifier of the first network device, wherein the status data is the data format supported by the first network device; and

the conversion module is configured to: convert the status data into the standard data format based on the first correspondence, and send converted status data to the control device.

10. The apparatus according to claim 7, wherein the receiving module is configured to receive a device access request of the first network device, wherein the device access request carries a model number of the first network device and the device identifier of the first network device;

the determining module is configured to determine, based on the model number of the first network device, a first digital twin model corresponding to the model number of the first network device; and

the apparatus further comprises an establishing module, configured to establish a third correspondence between the device identifier of the first network device and the first digital twin model.

11. The apparatus according to claim 10, wherein the first correspondence comprises a correspondence between a digital twin metamodel of the first digital twin model and a standard metamodel of the standard model; and

the conversion module is configured to: determine one or more standard metamodels of the upper device type based on the correspondence between a digital twin metamodel of the first digital twin model and a standard metamodel of the standard model; and convert, based on the first digital twin model and the one or more standard metamodels of the upper device type, the first control entity into the data format supported by the first network device.

12. The apparatus according to any one of claims 7 to 11, wherein the first digital twin model comprises one or more digital twin metamodels, and the digital twin metamodel comprises one or more of the following: a property metamodel, a command metamodel, an event metamodel, a data metamodel, a component metamodel, and a relationship metamodel.

13. A server, wherein the server comprises a communication bus, at least one network interface, a memory, and at least one processor, the communication bus is used to perform information transmission between the foregoing components, the at least one network interface is configured to communicate with another device or a communication network, the memory stores program code, and the at least one processor implements the device control method according to any one of claims 1 to 6 by reading and executing the program code stored in the memory.

14. A computer-readable storage medium, wherein when program code in the computer-readable storage medium is executed by a processor of a server, the server is enabled to perform the device control method according to any one of claims 1 to 6.