



**RCS-2000 API**

## Legal Information

**© 2021 Hangzhou Hikrobot Technology Co., Ltd. All rights reserved.**

This Document (hereinafter referred to be "the Document") is the property of Hangzhou Hikrobot Digital Technology Co., Ltd. or its affiliates (hereinafter referred to as "Hikrobot"), and it cannot be reproduced, changed, translated, or distributed, partially or wholly, by any means, without the prior written permission of Hikrobot. Unless otherwise expressly stated herein, Hikrobot does not make any warranties, guarantees or representations, express or implied, regarding to the Document, any information contained herein.

### **LEGAL DISCLAIMER**

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, THE DOCUMENT IS PROVIDED "AS IS" AND "WITH ALL FAULTS AND ERRORS". HIKROBOT MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT. IN NO EVENT WILL HIKROBOT BE LIABLE FOR ANY SPECIAL, CONSEQUENTIAL, INCIDENTAL, OR INDIRECT DAMAGES, INCLUDING, AMONG OTHERS, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION OR LOSS OF DATA, CORRUPTION OF SYSTEMS, OR LOSS OF DOCUMENTATION, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), OR OTHERWISE, IN CONNECTION WITH THE USE OF THE DOCUMENT, EVEN IF HIKROBOT HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES OR LOSS.

# Contents

<b>Chapter 1 Overview .....</b>	<b>1</b>
1.1 Introduction .....	1
1.2 Update History .....	1
1.3 Notice .....	2
<b>Chapter 2 API Description .....</b>	<b>4</b>
2.1 API Format .....	4
2.2 Operation Method .....	5
2.3 Message Format .....	5
2.4 Others .....	6
<b>Chapter 3 Security .....</b>	<b>7</b>
3.1 Authentication .....	7
<b>Chapter 4 Typical Application .....</b>	<b>8</b>
4.1 AMR Carries Rack and Leaves without Rack .....	8
4.2 AMR Carries Rack and Backs with Rack .....	8
4.3 Roller AMR Receives Materials .....	10
<b>Chapter 5 API Reference .....</b>	<b>11</b>
5.1 API List .....	11
5.2 API Provided by RCS-2000 .....	12
5.2.1 genAgvSchedulingTask .....	12
5.2.2 continueTask .....	15
5.2.3 cancelTask .....	17
5.2.4 setTaskPriority .....	19
5.2.5 bindPodAndBerth .....	20
5.2.6 bindPodAndMat .....	22
5.2.7 bindCtnrAndBin .....	23
5.2.8 lockPosition .....	25

5.2.9 syncMapDatas .....	26
5.2.10 queryPodBerthAndMat .....	29
5.2.11 queryTaskStatus .....	31
5.2.12 queryAgvStatus .....	33
5.2.13 stopRobot .....	35
5.2.14 resumeRobot .....	36
5.2.15 setAreaState .....	38
5.2.16 blockArea .....	39
5.2.17 genPreScheduleTask .....	41
5.3 API Provided by Third-Party .....	42
5.3.1 agvCallback .....	42
5.3.2 warnCallback .....	44
5.3.3 bindNotify .....	46
<b>Appendix A. Status Code .....</b>	<b>49</b>
<b>Appendix B. AMR Status .....</b>	<b>50</b>

# Chapter 1 Overview

## 1.1 Introduction

This manual provides some open APIs designed on RESTful style for the third-party platform to connect to Robot Control System (hereafter referred as to "RCS-2000") and control Automatic Mobile Robots (AMRs) for logistics scheduling in the factory. Some typical applications developed by these open APIs are also provided for reference.

The RCS-2000 is a logistics scheduling system used in the smart factory, it can generate and assign tasks to different kinds of AMRs to carry the racks, materials, goods, and so on, according to the task conditions. It can also monitor the task executing status to processing the exception situations in time, which makes sure the regular and accurate working of AMRs.

## 1.2 Update History

### Summary of Changes in Version 3.1.4\_May/2021

Version	Summary of Changes
Version 3.1.4_May/2021	1. Edited the request message of API <b>genAgvSchedulingTask</b> : deleted one field <b>sceneTyp</b> ; added 3 location types to the field <b>type</b> of <b>positionCodePath</b> : "07" (container ID), "08" (roadway strategy), and "09" (roadway area).
	2. Edited the request message of API <b>bindPodAndBerth</b> : added one field <b>characterValue</b> (trait value).
	3. Added one API for binding or unbinding a container and a bin: <b>bindCtnrAndBin</b> .
	4. Added one API for emptying or unblocking the specified area: <b>blockArea</b> .
	5. Added one API for sending the pre-schedule: <b>genPreScheduleTask</b> .
	6. Added one API for notifying the third-party platform of the binding or unbinding operation: <b>bindNotify</b> .

**Summary of Changes in Version 3.1.0\_July/2020**

Version	Summary of Changes
Version 3.1.0_July/2020	1. Edited the API absolute address to <code>"/rcms/services/rest/hikRpcService/[apiName]"</code> .
	2. Edited the default port No. in the API to 8182.
	3. Updated the API <b><i>genAgvSchedulingTask</i></b> : added the forklift function.
	4. Extended the request message of API <b><i>cancelTask</i></b> : added one parameter <b><i>forceCancel</i></b> (task cancel mode).
	5. Extended the request message of API <b><i>bindPodAndBerth</i></b> : added one parameter <b><i>podDir</i></b> (rack direction).
	6. Added one API for stopping the specified AMR or all AMRs: <b><i>stopRobot</i></b> .
	7. Added one API for resuming the AMR: <b><i>resumeRobot</i></b> .
	8. Added one API for blocking or unblocking the area: <b><i>setAreaState</i></b> .
	9. Edited the API for searching for the AMR status <b><i>queryAgvStatus</i></b> : edited the request URL to <code>"http://[address]:8083/rcms-dps/rest/queryAgvStatus"</code> .

**Summary of Changes in Version 2.5\_May/2020**

Version	Summary of Changes
Version 2.5_May/2020	1. Edited the API absolute address to <code>"/cms/services/rest/hikRpcService/[apiName]"</code> .
	2. Deleted one API for sending data synchronization requirement to the third-party platform when the information has changed: <code>syncNotify</code> .

**Summary of Changes in Version 2.2.3\_June/2019**

New document.

**1.3 Notice**

- When the RCS-2000 calls the APIs provided by the third-party platform, the default connection timeout threshold is 30 seconds, and the default returning timeout threshold is 60 seconds. If the timeout is longer than the default thresholds, the RCS-2000 will return the information of connection failed.

If connecting to the third-party platform failed, you can try again after five seconds, and by default, up to 5 failure connection attempts are allowed.

- When calling an API, make sure the **reqCode** (request IDs) in the request and response message are same.
- If the third-party platform is developed by C# or JAVA language, we have provided demos to quickly start, please contact our technical supports to get the demos.
- The exhaustive request and response parameters are more than that introduced in this manual, the third-party platform can choose required parameters according to the business.
- REST (REpresentational State Transfer) is a protocol design method which abstracts all information as the resources. The abstracted resources are marked by the uniform identifies, i.e., URI (Uniform Resource Identifiers) for simple and extendable management.

## Chapter 2 API Description

### 2.1 API Format

The APIs in this manual are all in URL format, which defines and provides a unique address for resources to access and implement different functions.

The detailed API format definition is shown below:

```
<protocol>://[address][:port][abs_path]
```

#### **protocol**

Protocol type that designing APIs based on, in this manual, the protocol type is "http".

#### **address**

Domain name or IP address of network device.

#### **port**

Port No.: for web server, the default port No. is 8182.

#### **abs\_path**

An absolute address to define a resource, you can connect to and operate the resource via this address. It varies according to different platform.

- For RCS-2000, the absolute address of its resources is "/rcms/services/rest/hikRpcService/[apiName]".
- For the third-party platform, the absolute address of its resources is "/xxx/[apiName]".



#### **Note**



- The [apiName] in the absolute address is used to distinguish the resources and functions, such as genAgvSchedulingTask, whose function is to generate task.
  - To simplify the description in this manual, we use **apiName** to replace the complete API format.
  - For API of searching for AMR status, the request URL is "http://[address][:port]/rcms-dps/rest/queryAgvStatus", and the port No. is 8083.
- 

### **Base Access Address**

To simplify API calling, you can define the path before **apiName** as the base access address **baseURL**. See the table below for details:

Platform	baseURL
RCS-2000	http://IP:PORT/rcms/services/rest/hikRpcService



Platform	baseURL
	 <b>Note</b> The default web server port No. is 8182.
RCS-2000 (AMR status search)	http://IP:PORT/rcms-dps/rest  <b>Note</b> The default port No. is 8083.

## 2.2 Operation Method

To implement different functions of resources represented by each API, operation method is required. As the APIs in this manual is designed based on HTTP, the operation methods are same as that supported by HTTP.

Method	Description
POST	Create or add resources.
GET	Search or get resources.
PUT	Update or set resources.
DELETE	Delete resources.

### Note

In this manual, only the POST operation method is available.

---

## 2.3 Message Format

During the development based on the open APIs, the request and response message for communication and interaction is in JSON format, and the fields in the message are named by lower camel case.

JSON format is a subset of JAVA script, which is a lightweight data format, and this format can be quickly parsed. See the example below.

```
{
  "code": "0",
  "data": "F01169C808C317111G",
  "message": "successful",
  "reqCode": "468513"
}
```

## 2.4 Others

### Time Format

The time appeared in the interaction between device and system adopts ISO8601 format, that is, "YYYY-MM-DD hh:mm:ss". For example, 2019-06-01 08:30:00.

### Error Processing

When calling the open APIs, if error occurs, the response message will directly return the error code, you can get the error description and reason according to the returned response message. See **Status Code** (Appendix. A) for detailed error codes and description.

## Chapter 3 Security

### 3.1 Authentication

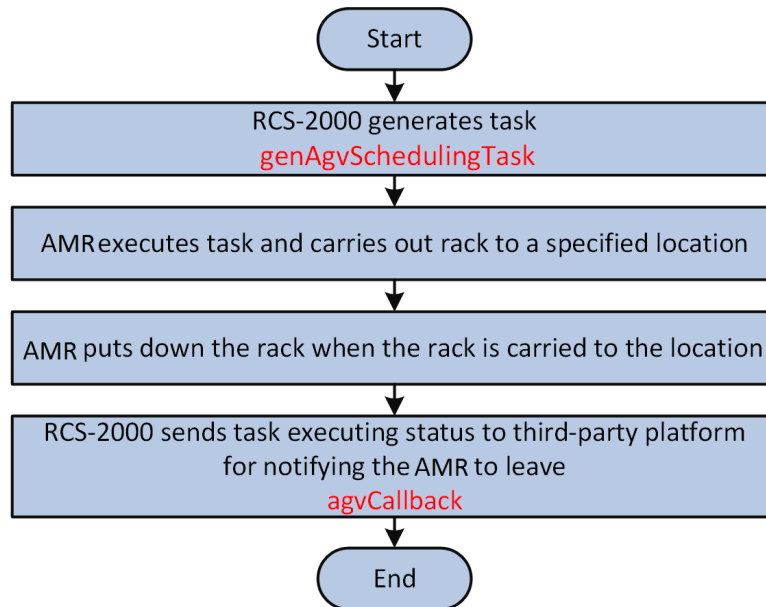
The authentication of the open API is based on token (**tokenCode**) transmitted during request and response. The token is a string generated by Hikrobot system and will be transmitted to the third-party platform for authentication when calling APIs.

## Chapter 4 Typical Application

### 4.1 AMR Carries Rack and Leaves without Rack

In this application scene, the AMR carries a rack from the location A to a specific location B after the third-party platform calling an API to set task parameters and generate task. When arriving at location B, the AMR puts down the rack and directly leave. This application scene is usually available when the time consumption of processing goods on rack is long.

#### Steps



**Figure 4-1 Flow of AMR Carrying Rack and Leaving without Rack**

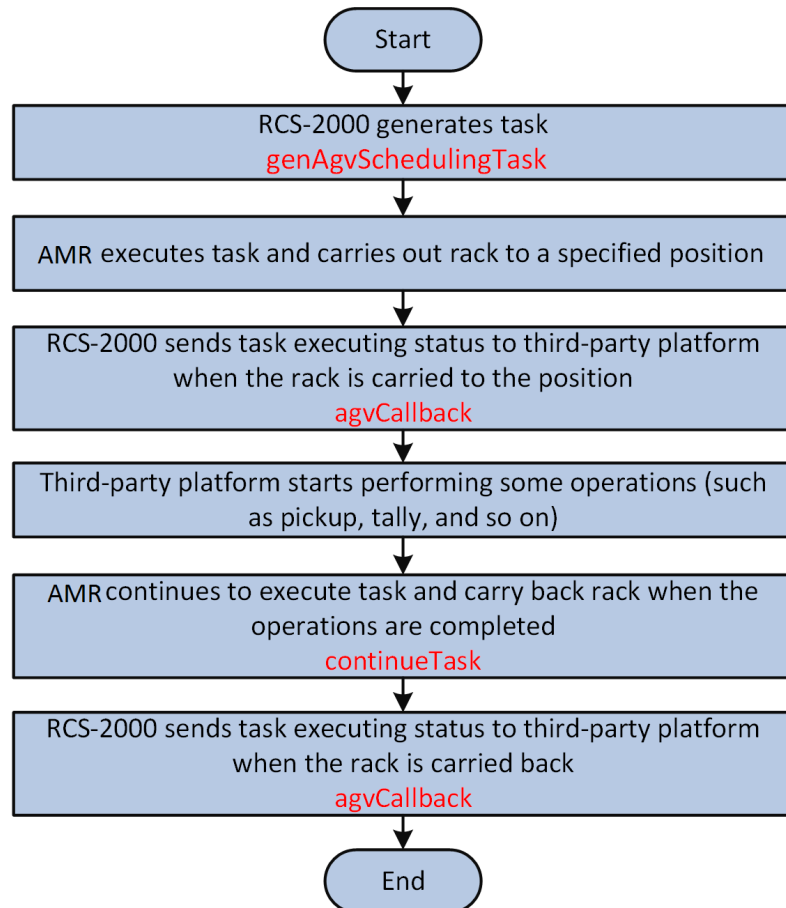
1. The third-party platform calls **genAgvSchedulingTask** to generate task via the RCS-2000.  
The AMR starts executing the task and carrying a rack to a specified location.
2. The AMR puts down the rack when it arrived at the location.
3. The RCS-2000 calls **agvCallback** to send task executing status to the third-party platform for notifying the AMR to leave.

### 4.2 AMR Carries Rack and Backs with Rack

In this application scene, the AMR carries a rack from the location A to a specific location B after the third-party platform calling an API to set task parameters and generate task. When arriving at location B, the AMR keeps carrying the rack and waits until the third-party platform call an API to continue the task. And then the AMR carries back the rack whose goods has been processed to

location A. This application scene is usually available when the time consumption of processing goods on rack is short.

### Steps



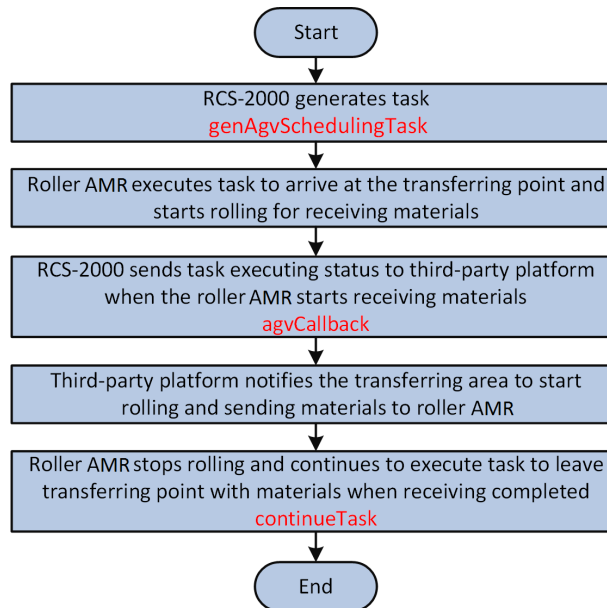
**Figure 4-2 Flow of AMR Carrying Rack and Backing with Rack**

1. The third-party platform calls *genAgvSchedulingTask* to generate task via the RCS-2000.  
The AMR starts executing the task and carrying out a rack to a specified location.
2. The RCS-2000 calls *agvCallback* to send task executing status to the third-party platform when the rack is carried to the location.
3. The third-party platform starts performing some operations, such as pickup, tally, and so on.
4. The third-party platform calls *continueTask* when all operations are completed to continue executing task.  
The AMR carries back the rack.
5. The RCS-2000 calls *agvCallback* to send task executing status to the third-party platform when the rack is carried back.

### 4.3 Roller AMR Receives Materials

The roller AMR is mainly applied to improve the automatic level of factory logistics. It can reduce the length of conveyor belt during long distance transportation; for the workshop which can not use forklift, it can convey very heavy material. Here introduces the progress of controlling roller AMR to complete the task of receiving materials.

#### Steps



**Figure 4-3 Flow of Roller AMR Receiving Materials**

1. The third-party platform calls **genAgvSchedulingTask** to generate task via the RCS-2000.  
The roller AMR executes the task, arrives at the transferring point, and starts rolling to get ready for receiving materials.
2. The RCS-2000 calls **agvCallback** to send the task executing status to the third-party platform when the AMR is ready for receiving materials.
3. The third-party platform notifies the transferring area to start rolling and sending materials to the roller AMR.
4. The third-party platform calls **continueTask** to continue executing task when the materials is received by roller AMR..  
The roller AMR stops rolling and leaves the transferring point with received materials.

## Chapter 5 API Reference

### 5.1 API List

#### List of Commonly Used APIs

Function	API Name	Provider
Create a task	<i>genAgvSchedulingTask</i>	RCS-2000
Continue executing the next sub task.	<i>continueTask</i>	RCS-2000
Cancel a task	<i>cancelTask</i>	RCS-2000
Send the task executing status to third-party platform	<i>agvCallback</i>	Third-party Platform

#### List of Optional APIs

Function	API Name	Provider
Set the task priority	<i>setTaskPriority</i>	RCS-2000
Bind and unbind the rack and location	<i>bindPodAndBerth</i>	RCS-2000
Bind and unbind the material batch and rack	<i>bindPodAndMat</i>	RCS-2000
Enable or disable the location	<i>lockPosition</i>	RCS-2000
Synchronize map information with that in RCS-2000	<i>syncMapDatas</i>	RCS-2000
Search relations among rack, location, and material batch	<i>queryPodBerthAndMat</i>	RCS-2000
Search for the task executing status	<i>queryTaskStatus</i>	RCS-2000
Search for the AMR status	<i>queryAgvStatus</i>	RCS-2000
Stop AMR(s)	<i>stopRobot</i>	RCS-2000
Resume a AMR	<i>resumeRobot</i>	RCS-2000
Block or unblock a area	<i>setAreaState</i>	RCS-2000
Bind or unbind the container and bin	<i>bindCtnrAndBin</i>	RCS-2000
Empty or unblock the specified area	<i>blockArea</i>	RCS-2000
Send the pre-schedule	<i>genPreScheduleTask</i>	RCS-2000

Function	API Name	Provider
Send the alarm to the third-party platform	<b><i>warnCallback</i></b>	Third-party Platform
Notify the third-party platform of the binding or unbinding operation	<b><i>bindNotify</i></b>	Third-party Platform

## 5.2 API Provided by RCS-2000

### 5.2.1 genAgvSchedulingTask

Robot Control System (RCS) creates a task and the third-party platform applies the task to AMR.

#### API Definition

**Table 5-1 POST [http://\[address\]\[:port\]/rcms/services/rest/hikRpcService/genAgvSchedulingTask](http://[address][:port]/rcms/services/rest/hikRpcService/genAgvSchedulingTask)**

<b>API Name</b>	genAgvSchedulingTask				
<b>Function</b>	Robot Control System (RCS) creates a task and the third-party platform applies the task to AMR.				
<b>Protocol</b>	REST				
<b>Provider</b>	RCS-2000				
<b>Caller</b>	Third-party platform				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	taskTyp	String	16	Req.	Task type, which is the same as the major task type configured in RCS-2000: <ul style="list-style-type: none"><li>Build-in task type: "F01"-carry and transfer rack,</li></ul>



					<p>"F02"-empty/full rack exchange,  "F03"-carry and transfer by CMR,  "F04"-rack outbound,  "F05"-rotate rack,  "F06"-elevator task.</p> <ul style="list-style-type: none"> <li>Fork lift task type:  "F11"-transfer from high bay rack to workstation,  "F12"-transfer from workstation to high bay rack,  "F13"-transfer from roadway to workstation,  "F14"-transfer from workstation to roadway,  "F15"-shuttle from high bay rack to workstation,  "F16"-shuttle from workstation to high bay rack,  "F17"-shuttle from roadway to workstation,  "F18"-shuttle from workstation to roadway,  "F20"-cross-floor forklift main task.</li> </ul>
	ctnrTyp	String	16	Opt.	Container type, this field is required for forklift/CTU (carton transport unit) task.
	ctnrCode	String	32	Opt.	Container ID, this field is required for forklift/CTU (carton transport unit) task.
	wbCode	String	32	Opt.	Workstation ID, which consists of letters and digits, and it must be same with that configured by RCS-2000.
	positionCodePath	Object[]	50	Opt.	<p>Rack moving pattern, which consists of multiple locations (up to 50 locations are allowed).</p> <p><b>positionCode:</b> Location ID, which is predefined with detailed coordinate information on map; the maximum string length is 64 bytes.</p> <p><b>type:</b> Location types:  "00"-actual location on map,  "01"-location of a specific material batch,  "02"-available location of area selection strategy,</p>

					"03"-rack No., "04"-available location in an area, "05"-bin ID (for forklift/CTU only), "06"-roadway ID, "07"-container ID, "08"-roadway strategy, "09"-roadway area
podCode	String	16	Opt.		Rack ID, it can be empty if no rack specified.
podDir	String	4	Opt.		Rack directions: "180"-leftward, "0"-rightward, "90"-upward, "-90"-downward; it can be empty if no direction specified.
podTyp	String	16	Opt.		Rack types: "1"-empty rack of all types (default), "2"-type of rack linked with configured workstation (if the linked rack is empty, it also represents empty rack of all types), other values-empty rack with a specific type. Rack type No.: empty rack of the specified type.
materialLot	String	32	Opt.		Material batch ID. You can set <b>materialLot</b> and <b>podCode</b> at same time to bind the rack with material lot, or you can set <b>materialLot</b> and <b>wbCode</b> to bind the material lot with the rack at the specified workstation. For roadway tasks, this field is used for configuring the material trait value.
priority	String	3	Opt.		Task priority, range: [1,127], and larger number corresponds to higher priority. If it is not configured, the task template priority takes effect.
taskCode	String	64	Opt.		Task ID, if it is not configured, the system will generate automatically.
agvCode	String	16	Opt.		AMR ID, if it is not configured, the RCS-2000 will automatically select an optimal AMR to execute the task.

	data	String	2000	Opt.	Custom content, up to 2000 characters are allowed.
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
	data	String	2000	Opt.	Custom content to be returned, such as task ID.
<b>Remarks</b>	One of <b>wbCode</b> and <b>positionCodePath</b> must be configured to confirm the location information in task. If you want to specify multiple locations in the task, e.g., start location and end location, the field <b>positionCodePath</b> should be configured.				
<b>Sample</b>	<b>Request</b>	<pre>{   "reqCode": "468513",   "taskTyp": "F01",   "positionCodePath": [{     "positionCode": "p01",     "type": "00"   }],   {     "positionCode": "x02",     "type": "02"   }],   "podCode": "100001",   "podDir": "0",   "priority": "1" }</pre>			
	<b>Response</b>	<pre>{   "code": "0",   "data": "F01169C808C317111G",   "message": "successful",   "reqCode": "468513" }</pre>			

### 5.2.2 continueTask

Continue executing the next sub task.

## API Definition

Table 5-2 POST http://[address][:port]/rcms/services/rest/hikRpcService/continueTask

<b>API Name</b>	continueTask				
<b>Function</b>	Continue executing the next sub task.				
<b>Protocol</b>	REST				
<b>Provider</b>	RCS-2000				
<b>Caller</b>	Third-party platform				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	wbCode	String	32	Opt.	Workstation ID, it must be same with that configured by RCS-2000.
	podCode	String	16	Opt.	Rack ID
	agvCode	String	16	Opt.	AMR ID, if it is not configured, the RCS-2000 will automatically select an optimal AMR to execute the task.
	taskCode	String	64	Opt.	Task ID (UUID). If it is not configured, it will be automatically generated by RCS-2000, and it is 64 bits UUID.
	taskSeq	String	32	Opt.	Sub task No. to be specified to continue executing. If this field is not configured, by default, the next sub task will be executed.
	nextPositionCode	Object	40	Opt.	The location information of sub tasks. This field is required when the task type is externally configured. <b>positionCode</b> : position ID, which is predefined with detailed coordinate information on map.

					<b>type:</b> location types: "00"-actual location on map, "02"-strategy
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful"
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
<b>Remarks</b>	Only one of <b>wbCode</b> , <b>agvCode</b> , <b>taskCode</b> , and <b>podCode</b> should be configured at one time, and the parameter to be configured depends on the trigger type configured in the task template. The parameter <b>taskCode</b> is recommended.				
<b>Sample Codes</b>	<b>Request</b>	<pre>{   "reqCode": "123",   "taskCode": "123456",   "nextPositionCode": {     "positionCode": "p02",     "type": "00"   } }</pre>			
	<b>Response</b>	<pre>{   "code": "0",   "message": "successful",   "reqCode": "123" }</pre>			

### 5.2.3 cancelTask

Cancel the task that is executing or is generated but waiting for being executed.

#### API Definition

**Table 5-3 POST http://[address][:port]/rcms/services/rest/hikRpcService/cancelTask**

<b>API Name</b>	cancelTask
<b>Function</b>	Cancel the task that is executing or is generated but waiting for being executed.
<b>Protocol</b>	REST
<b>Provider</b>	RCS-2000
<b>Caller</b>	Third-party platform

Request	Parameter	Data Type	Max. Length	Req. or Opt.	Description
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	forceCancel	String	16	Opt.	Task cancel mode: "0" (default)-AMR puts down the rack at current location and its status turns to "idle", "1"-AMR carries the rack and returns back, this mode is supported by LMR (latent mobile robot) only.
	matterArea	String	16	Opt.	Inbound area No., it is valid only when the value of <b>forcecancel</b> is "1"; if this field is not configured, the inbound area is the warehouse area.
	agvCode	String	16	Opt.	AMR ID, whose executing task will be canceled.
	taskCode	String	64	Opt.	ID (UUID) of task to be canceled.
Response	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
Remarks	One of <b>taskCode</b> and <b>agvCode</b> must be configured to confirm the task to be canceled or the AMR to be freed. If both the two parameters are configured, only the value of <b>agvCode</b> will take effect.				
Sample Codes	Request	<pre>{   "reqCode": "1541954B96B1112",   "forceCancel": "1",   "matterArea": "abc",</pre>			

		<pre>"taskCode": "123456" }</pre>
	<b>Response</b>	<pre>{   "code": "0",   "message": "successful",   "reqCode": "1541954B96B1112" }</pre>

### Remarks

- If an AMR is executing a task and carrying a rack, when the task is canceled and the cancel mode is "0", the AMR will put down the rack at any location, and its status turns to "idle"; when the cancel mode is "1", the AMR will carry the rack and execute inbound, if there is no space of inbound area, the error will be returned and canceling failed.
- Forklift only supports the cancel mode "0".

### 5.2.4 setTaskPriority

Set the task priority from 1 to 127, and the larger the value, the higher the priority. The priority takes effect only when the AMRs are not enough and there are multiple tasks with different priorities. The tasks will be assigned to the AMR according to priority order.

### API Definition

**Table 5-4 POST [http://\[address\]\[:port\]/rcms/services/rest/hikRpcService/setTaskPriority](http://[address][:port]/rcms/services/rest/hikRpcService/setTaskPriority)**

<b>API Name</b>	setTaskPriority				
<b>Function</b>	Set task priority from 1 to 127, and the larger the value, the higher the priority.				
<b>Protocol</b>	REST				
<b>Provider</b>	RCS-2000				
<b>Caller</b>	Third-party platform				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS

	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	priorities	List			
	taskCode	String	64	Req.	Task ID (UUID)
	priority	String	32	Req.	Task priority, range: [1,127], the larger the value, the higher the priority.
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful"
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
	data	String	2000	Opt.	Custom content to be returned, such as task ID.
<b>Remarks</b>	You can set the priority for the tasks that are not assigned to the AMR only; If the task has been assigned to AMR, it is invalid for setting task priority.				
<b>Sample Codes</b>	<b>Request</b>	<pre>{   "reqCode": "1234567",   "priorities": [{     "priority": "1",     "taskCode": "1232"   },   {     "priority": "2",     "taskCode": "3214"   }] }</pre>			
	<b>Response</b>	<pre>{   "code": "0",   "message": "successful",   "reqCode": "1234567" }</pre>			

### 5.2.5 bindPodAndBerth

Bind or unbind the rack and location.



## API Definition

Table 5-5 POST [http://\[address\]\[:port\]/rcms/services/rest/hikRpcService/bindPodAndBerth](http://[address][:port]/rcms/services/rest/hikRpcService/bindPodAndBerth)

<b>API Name</b>	bindPodAndBerth				
<b>Function</b>	Bind or unbind a rack and one location.				
<b>Protocol</b>	REST				
<b>Provider</b>	RCS-2000				
<b>Caller</b>	Third-party platform				
<b>Remarks</b>	When unbinding, to avoid misoperation, both parameters <b>podCode</b> and <b>positionCode</b> should be configured for verification.				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	podCode	String	16	Req.	Rack ID
	positionCode	String	32	Req.	Position ID, which is predefined with detailed coordinate information on map, and it is a unique string configured via RCS-2000.
	podDir	String	6	Opt.	Rack direction: "0"-horizontal (default), "1"-vertical. If this field is not configured, the rack direction is horizontal by default.
	characterValue	String	64	Opt.	Trait value (only for racks on the roadway)
	indBind	String	1	Req.	Bind or unbind: "1"-bind, "0"-unbind.
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful".

	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
Sample Codes	Request	{ "reqCode": "12345678", "podCode": "100001", "positionCode": "p05", "podDir": "0", "indBind": "1" }			
	Response	{ "code": "0", "message": "successful", "reqCode": "12345678" }			

### 5.2.6 bindPodAndMat

Bind and unbind the material batch and rack.

#### API Definition

Table 5-6 POST [http://\[address\]\[:port\]/rcms/services/rest/hikRpcService/bindPodAndMat](http://[address][:port]/rcms/services/rest/hikRpcService/bindPodAndMat)

API Name	bindPodAndMat				
Function	Bind and unbind material batch and rack.				
Protocol	REST				
Provider	RCS-2000				
Caller	Third-party platform				
Remarks	When unbinding, to avoid misoperation, both parameters <b>podCode</b> and <b>materialLot</b> should be configured for verification.				
Request	Parameter	Data Type	Max. Length	Req. or Opt.	Description
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS

	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	podCode	String	16	Req.	Rack ID
	materialLot	String	32	Req.	Material batch ID
	indBind	String	1	Req.	Bind or unbind: "1"-bind, "0"-unbind.
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
	data	String	2000	Opt.	Custom content to be returned, such as task ID.
<b>Sample Codes</b>	<b>Request</b>	<pre>{   "reqCode": "1541954B96B1112",   "podCode": "100001",   "materialLot": "123",   "indBind": "1" }</pre>			
	<b>Response</b>	<pre>{   "code": "0",   "data": "",   "message": "successful",   "reqCode": "1541954B96B1112" }</pre>			


### 5.2.7 bindCtnrAndBin


Bind or unbind the container and bin. For CTU (carton transport unit) and forklift tasks, you can call this API to write in the container type and ID to the bin.

#### Request URL Definition

**Table 5-7 POST [http://\[address\]\[:port\]/rcms/services/rest/hikRpcService/bindCtnrAndBin](http://[address][:port]/rcms/services/rest/hikRpcService/bindCtnrAndBin)**

<b>API Name</b>	bindCtnrAndBin
<b>Function</b>	Bind or unbind the container and bin.
<b>Protocol</b>	REST

<b>Provider</b>	RCS-2000				
<b>Caller</b>	Third-party platform				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	ctnrCode	String	16	Req.	Container ID
	ctnrTyp	String	16	Req.	Container type
	stgBinCode	String	32	Opt.	Bin ID.  <b>Note</b> One of <b>stgBinCode</b> and <b>positionCode</b> is required.
	binName	String	32	Opt.	Custom bin ID, which will be verified by the RCS-2000. For example, when the system configuration number 10110 is set to true, it indicates that the custom bin ID and bin ID should be one-to-one, if one custom bin ID corresponds to multiple bin ID, an error will be reported.
	characterValue	String	64	Opt.	Trait value (only for the forklift roadway)
	positionCode	String	32	Req.	Position ID, which is predefined with detailed coordinate information on the map, and it is a unique string configured via RCS-2000. This field is used for binding or unbinding containers and the bin of virtual racks.

					 <b>Note</b> One of <b>stgBinCode</b> and <b>positionCode</b> is required.
	indBind	String	1	Req.	Bind or unbind: "1"-bind, "0"-unbind.
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	data	String	2000	Opt.	Custom content to be returned.
	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
<b>Sample Codes</b>	<b>Request</b>	<pre>{   "reqCode": "12345678",   "ctnrTyp": "C1",   "stgBinCode": "p05",   "indBind": "1" }</pre>			
	<b>Response</b>	<pre>{   "code": "0",   "message": "successful",   "reqCode": "12345678" }</pre>			

## 5.2.8 lockPosition

Enable or disable the location. When the location is disabled, it cannot be found in the area.

### API Definition

**Table 5-8 POST [http://\[address\]\[:port\]/rcms/services/rest/hikRpcService/lockPosition](http://[address][:port]/rcms/services/rest/hikRpcService/lockPosition)**

<b>API Name</b>	lockPosition
<b>Function</b>	Enable or disable the location. When the location is disabled, it cannot be found in the area.
<b>Protocol</b>	REST
<b>Provider</b>	RCS-2000
<b>Caller</b>	Third-party platform

Request	Parameter	Data Type	Max. Length	Req. or Opt.	Description
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	positionCode	String	32	Req.	Position ID, which is predefined with detailed coordinate information on map, and it is a unique string configured via RCS-2000.
	indBind	String	1	Req.	Enable or disable: "1"-enable, "0"-disable.
Response	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
	data	String	2000	Opt.	Custom content to be returned, such as task ID.
Sample Codes	Request	<pre>{   "reqCode": "1541954B96B1112",   "positionCode": "p02",   "indBind": "1" }</pre>			
	Response	<pre>{   "code": "0",   "message": "successful",   "reqCode": "1541954B96B1112" }</pre>			


### 5.2.9 syncMapDatas

Synchronize map information with that in RCS-2000.

## API Definition

Table 5-9 POST http://[address][:port]/rcms/services/rest/hikRpcService/syncMapDatas

<b>API Name</b>	syncMapDatas				
<b>Function</b>	Synchronize map information with that in RCS-2000.				
<b>Protocol</b>	REST				
<b>Provider</b>	RCS-2000				
<b>Caller</b>	Third-party platform				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	mapDataCode	String	32	Opt.	Unique location code on map
	mapShortName	String	32	Req.	Alias of the map that needs to be synchronized
	dataTyp	String	6	Opt.	Map element type, when this field is not configured, all location codes of the map will be synchronized.
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
	data	List			
	cooX	String	8	Req.	X-coordinate of location code, unit: mm
	cooY	String	8	Req.	Y-coordinate of location code, unit: mm

	dataTyp	String	2	Req.	Map element types: "1"-storage section, "10"-workstation, "11"-charge station, "20"-interim storage area, "55"-roadway storage area
	direction	String	8	Opt.	Working station direction, in which a staff faces a rack to pickup: "180"-leftward, "0"-rightward, "90"-upward, "90"-downward.
	mapCode	String	16	Req.	Map ID
	mapDataCode	String	32	Req.	Unique location code on map
	positionCode	String	32	Req.	Position ID, which is predefined with detailed coordinate information on map, and it is a unique string configured via RCS-2000.
	berthType	String	2	Opt.	Storage section types: "1"-outer storage section, "2"-inner storage section, "3"-normal storage section.  <div>  <b>Note</b>            This field is required when the value of <b>dataTyp</b> is 1.         </div>
Sample Codes	Request	<pre>{   "reqCode": "1541954B96B1112",   "reqTime": "",   "clientCode": "",   "tokenCode": "",   "mapDataCode": "xxxxxxx",   "mapShortName": "xxxxxxx",   "dataTyp": "" }</pre>			
	Response	<pre>{   "code": "0",   "message": "successful",   "reqCode": "1541954B96B1112",   "data": [{     "berthType": "3",     "cooX": "17000.0",     "cooY": "18000.0",     "dataTyp": "1",     "direction": "0",     "mapCode": "AA",     "mapDataCode": "011724AA012414",</pre>			



```

    "positionCode": "011724AA012414"
  },
  {
    "berthType": "3",
    "cooX": "11000.0",
    "cooY": "21999.0",
    "dataTyp": "10",
    "direction": "0",
    "mapCode": "AA",
    "mapDataCode": "007586AA015172",
    "positionCode": "104"
  }
]
}

```

### 5.2.10 queryPodBerthAndMat

Search the relations among rack, location, and material batch.

#### API Definition

**Table 5-10 POST [http://\[address\]\[:port\]/rcms/services/rest/hikRpcService/queryPodBerthAndMat](http://[address][:port]/rcms/services/rest/hikRpcService/queryPodBerthAndMat)**

<b>API Name</b>	queryPodBerthAndMat				
<b>Function</b>	Search the relations among rack, location, and material batch.				
<b>Protocol</b>	REST				
<b>Provider</b>	RCS-2000				
<b>Caller</b>	Third-party platform				
<b>Remarks</b>	At least one of the following parameters should be configured: <b>podCode</b> , <b>materialLot</b> , <b>positionCode</b> , and <b>mapShortName</b> .				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.

## RCS-2000 API

	podCode	String	16	Opt.	Rack ID
	materialLot	String	32	Opt.	Material batch ID
	positionCode	String	16	Opt.	Position ID, which is predefined with detailed coordinate information on map, and it is a unique string configured via RCS-2000.
	areaCode	String	16	Opt.	Area ID
	mapShortName	String	16	Opt.	Map alias
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
	data	Object			
	areaCode	String	16	Opt.	Area ID
	materialLot	String	64	Opt.	Material batch ID
	podCode	String	16	Req.	Rack ID
	mapDataCode	String	32	Req.	Unique location code on map
	positionCode	String	32	Req.	Position ID, which is predefined with detailed coordinate information on map, and it is a unique string configured via RCS-2000.
<b>Sample Codes</b>	<b>Request</b>	<pre>{   "reqCode": "1541954B96B1110",   "mapShortName": "test" }</pre>			
	<b>Response</b>	<pre>{   "code": "0",   "message": "successful",   "reqCode": "1541954B96B1110",   "data": [{     "podCode": "100001",     "mapDataCode": "P02",     "positionCode": "P02"   }], }</pre>			

```
{
  "podCode": "100002",
  "mapDataCode": "P03",
  "positionCode": "P03"
}]
}
```

### 5.2.11 queryTaskStatus

Search for the task executing status according to the task ID or AMR ID, or search for executing statuses of multiple tasks in batch.

#### API Definition

**Table 5-11 POST [http://\[address\]\[:port\]/rcms/services/rest/hikRpcService/queryTaskStatus](http://[address][:port]/rcms/services/rest/hikRpcService/queryTaskStatus)**

<b>API Name</b>	queryTaskStatus				
<b>Function</b>	Search for the task executing status according to the task ID or AMR ID, or search for executing statuses of multiple tasks in batch.				
<b>Protocol</b>	REST				
<b>Provider</b>	RCS-2000				
<b>Caller</b>	Third-party platform				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	taskCodes	String[ ]	64	Opt.	Array of task IDs, and at least one of <b>taskCodes</b> and <b>agvCode</b> should be configured.
	agvCode	String	16	Opt.	AMR ID, and at least one of <b>taskCodes</b> and <b>agvCode</b> should be configured.
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.

	data	Object		
	taskCode	String	64	Req. Task ID (UUID)
	taskTyp	String	16	Req. Task type
	taskStatus	String	2	Req. Task status: "0"-sending exception, "1"-created, "2"-executing, "3"-sending, "4"-canceling, "5"-canceled, "6"-resending, "9"-completed, "10"-interrupted. Common values are: "0", "1", "2", "5", "9".
	agvCode	String	16	Opt. AMR ID, this field exists when the task has been assigned to.
	message	String	64	Req. Returned status description, e.g., "successful".
	reqCode	String	64	Req. Request ID, which is the same with that in corresponding request message
<b>Sample Codes</b>	<b>Request</b>	<pre>{   "reqCode": "1541954B96B1110",   "taskCodes": ["123", "234"] }</pre>		
	<b>Response</b>	<pre>{   "code": "0",   "message": "successful",   "reqCode": "1541954B96B1110",   "data": [{     "taskCode": "234",     "taskStatus": "2",     "taskTyp": "F01"   },   {     "taskCode": "123",     "taskStatus": "9",     "taskTyp": "F01"   } ]</pre>		


## 5.2.12 queryAgvStatus

Search for AMR status, including the AMR battery.

### API Definition

**Table 5-12 POST http://[address]:8083/rcms-dps/rest/queryAgvStatus**

<b>API Name</b>	queryAgvStatus				
<b>Function</b>	Search for AMR status, including the AMR battery.				
<b>Protocol</b>	REST				
<b>Provider</b>	RCS-2000				
<b>Caller</b>	Third-party platform				
<b>Remarks</b>	<ul style="list-style-type: none"> <li>API calling frequency: number of AMRs is less than 100: 5 seconds; number of AMRs is between 100 and 200: 10 seconds; number of AMRs is between 200 and 300: 15 seconds.</li> <li>The request URI is "http://[address]:8083/rcms-dps/rest/queryAgvStatus"</li> </ul>				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	mapShortName	String	32	Opt.	Alias of the map where the AMR locates.
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	data	Object			
	robotCode	String	16	Req.	AMR ID
	robotDir	String	4	Req.	AMR direction, range: [-180,360] degrees
	robotIp	String	64	Opt.	AMR IP address

	battery	String	4	Req.	AMR battery, range: [0,100]
	posX	String	8	Req.	AMR X-coordinate, unit: millimeter
	posY	String	8	Req.	AMR Y-coordinate, unit: millimeter
	mapCode	String	32	Req.	ID of map where the AMR locates
	speed	String	6	Req.	AMR current speed, unit: mm/s
	status	String	6	Req.	AMR status, see <b>AMR Status</b> for details.
	exclType	String	1	Req.	Whether the AMR is excluded: "1"-excluded, "0"-not excluded.   <b>Note</b> No task will be assigned to the excluded AMR.
	stop	String	1	Req.	Whether the AMR is stopped: "0"-no, "1"-yes.
	podCode	String	16	Opt.	Carried rack ID
	podDir	String	6	Opt.	Direction of move with rack
	path	String[ ]	300	Opt.	Task execution path, unit: millimeter, format: [x-coordinate, y-coordinate, direction], e.g., ["x,y,dir"], ["x,y,dir"], ["x,y,dir"]]
	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
<b>Sample Codes</b>	<b>Request</b>	<pre>{   "reqCode": "1541954B96B1112",   "reqTime": "2020-04-03 10:08:06",   "mapShortName": "test" }</pre>			
	<b>Response</b>	<pre>{   "code": "0",   "message": "successful",   "reqCode": "1541954B96B1112",   "data": [{     "robotCode": "1001",     "robotDir": "180",     "battery": "80",     "posX": "1.0",     "posY": "2.0",     "mapCode": "AA",     "speed": "100", </pre>			

```

        "status": "1",
        "exclType": "0",
        "stop": "1",
        "podCode": "200001",
        "podDir": "90",
        "path": [
            "[10000,20000,90]",
            "[20000,30000,-90]",
            "[20000,30000,180]",
            "[30000,40000,0]"
        ],
    },
    {
        "robotCode": "1001",
        "robotDir": "180",
        "battery": "80",
        "posX": "1.0",
        "posY": "2.0",
        "mapCode": "AA",
        "speed": "100",
        "status": "1",
        "exclType": "0",
        "stop": "1",
        "podCode": "200001",
        "podDir": "90"
    }
}

```


### 5.2.13 stopRobot

Stop the specified AMR or all AMRs.

#### API Definition

**Table 5-13 POST [http://\[address\]\[:port\]/rcms/services/rest/hikRpcService/stopRobot](http://[address][:port]/rcms/services/rest/hikRpcService/stopRobot)**

<b>API Name</b>	stopRobot				
<b>Function</b>	Stop the specified AMR or all AMRs.				
<b>Protocol</b>	REST				
<b>Provider</b>	RCS-2000				
<b>Caller</b>	Third-party platform				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>

	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	robotCount	String	64	Opt.	The number of AMRs to be stopped, "-1"-all AMRs.
	mapShortName	String	32	Opt.	Alias of the map where the AMR locates.  <b>Note</b> This field is required when the value of <b>robotCount</b> is "-1".
	robots	String[]	16	Opt.	List of AMR IDs.
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
<b>Sample Codes</b>	<b>Request</b>	<pre>{   "reqCode": "1541954B96B1112",   "robotCount": "2",   "robots": [     "1001",     "1002"   ] }</pre>			
	<b>Response</b>	<pre>{   "code": "0",   "message": "successful",   "reqCode": "1541954B96B1112" }</pre>			


### 5.2.14 resumeRobot

Resume the AMR, the resumed AMR will continue executing the uncompleted task.



## API Definition

Table 5-14 POST http://[address][:port]/rcms/services/rest/hikRpcService/resumeRobot

<b>API Name</b>	resumeRobot				
<b>Function</b>	Resume the AMR, the resumed AMR will continue executing the uncompleted task.				
<b>Protocol</b>	REST				
<b>Provider</b>	RCS-2000				
<b>Caller</b>	Third-party platform				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	robotCount	String	64	Opt.	The number of AMRs to be resumed, "-1"-all AMRs.
	mapShortName	String	32	Opt.	Alias of the map where the AMR locates.  <b>Note</b> This field is required when the value of <b>robotCount</b> is "-1".
	robots	String[ ]	16	Opt.	List of AMR IDs
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
<b>Sample Codes</b>	<b>Request</b>	<pre>{   "reqCode": "1541954B96B1112",   "robotCount": "2",</pre>			

		<pre> "robots": [   "1001",   "1002" ] } </pre>
	<b>Response</b>	<pre> {   "code": "0",   "message": "successful",   "reqCode": "1541954B96B1112" } </pre>

### 5.2.15 setAreaState

Block or unblock the specified area.

#### API Definition

**Table 5-15 POST [http://\[address\]\[:port\]/rcms/services/rest/hikRpcService/setAreaState](http://[address][:port]/rcms/services/rest/hikRpcService/setAreaState)**

<b>API Name</b>	setAreaState				
<b>Function</b>	Block or unblock the specified area.				
<b>Protocol</b>	REST				
<b>Provider</b>	RCS-2000				
<b>Caller</b>	Third-party platform				
<b>Remarks</b>	You can configure the area via RCS-2000				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	matterArea	String	16	Req.	ID of the area to be blocked or unblocked
	indBind	String	1	Req.	Block or unblock: "1"-block, "0"-unblock.

<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
<b>Sample Codes</b>	<b>Request</b>	<pre>{   "reqCode": "1541954B96B1112",   "matterArea": "2",   "indBind": "1" }</pre>			
	<b>Response</b>	<pre>{   "code": "0",   "message": "successful",   "reqCode": "1541954B96B1112" }</pre>			

### 5.2.16 blockArea

Empty or unblock the specified area. Emptying an area is to remove all objects from the area and block the area.

#### API Definition

**Table 5-16 POST [http://\[address\]\[:port\]/rcms/services/rest/hikRpcService/blockArea](http://[address][:port]/rcms/services/rest/hikRpcService/blockArea)**

<b>API Name</b>	blockArea				
<b>Function</b>	Empty or unblock the specified area. Emptying an area is to remove all objects from the area and block the area.				
<b>Protocol</b>	REST				
<b>Provider</b>	RCS-2000				
<b>Caller</b>	Third-party platform				
<b>Remarks</b>	<ul style="list-style-type: none"> <li>You can configure the area via RCS-2000</li> <li>When the area is emptied, the AMRs in the area leaves, and the AMRs to enter the area will bypass.</li> </ul>				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.

	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	matterArea	String	16	Req.	ID of the area to be emptied or unblocked
	indBind	String	1	Req.	Empty or unblock the area: "1"-empty, "0"-unblock.
	pause	String	1	Req.	Whether to pause emptying the area: "0" (no, default value) , 1 (yes).
	controlMod	String	2	Req.	Scheduling mode: "0" (move out of the area, default value), "1" (move to the interim park area), "2" (move to the specified area)
	targetArea	String	16	Opt.	The specified area ID. This field is required when the value of <b>controlMod</b> is "2".
	noticeThird	String	2	Req.	Whether to notify the third-party platform when the area is emptied: 0 (no), 1 (yes).
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
<b>Sample Codes</b>	<b>Request</b>	<pre>{   "reqCode": "1541954B96B1112",   "matterArea": "2",   "indBind": "1",   "pause": "0",   "controlMod": "2",   "targetArea": "HF002" }</pre>			
	<b>Response</b>	<pre>{   "code": "0",   "message": "successful",   "reqCode": "1541954B96B1112" }</pre>			

### 5.2.17 genPreScheduleTask

Robot Control System (RCS) applies the pre-schedule after verifying the request sent by the third-party platform. The pre-schedule is to notify RCS of the actual task in advance so that RCS can send a free AMR to the target position and the AMR is in standby for the upcoming actual task. It improves the scheduling efficiency.

#### API Definition

**Table 5-17 POST [http://\[address\]\[:port\]/rcms/services/rest/hikRpcService/genPreScheduleTask](http://[address][:port]/rcms/services/rest/hikRpcService/genPreScheduleTask)**

<b>API Name</b>	genAgvSchedulingTask				
<b>Function</b>	Robot Control System (RCS) applies the pre-schedule after verifying the request sent by the third-party platform.				
<b>Protocol</b>	REST				
<b>Provider</b>	RCS-2000				
<b>Caller</b>	Third-party platform				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	positionCode	String	32	Req.	Task position call sign
	nextTask	String	32	Req.	Pre-schedule time period. Remaining seconds before the actual task is generated: "-1" (clear the pre-schedule of a position).
	agvTyp	String	256	Req.	AMR type, with which the complete the actual task.
	priority	String	3	Opt.	Task priority, range: [1,127], and larger number corresponds to higher priority. If it is not configured, the task template priority takes effect.

	clearPreTask	String	2	Opt.	Whether to clear the previous pre-schedule: "0" (no, default value), 1 (yes).
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
	data	String	2000	Opt.	Custom content to be returned.
<b>Sample</b>	<b>Request</b>	<pre>{   "reqCode": "123",   "tokenCode": "128654",   "positionCode": "p01",   "nextTask": "20",   "agvTyp": "01",   "clearPreTask": "0" }</pre>			
	<b>Response</b>	<pre>{   "code": "0",   "data": "F01169C808C317111G",   "message": "successful",   "reqCode": "468513" }</pre>			

## 5.3 API Provided by Third-Party

### 5.3.1 agvCallback

Send the task executing status to the third-party platform.

#### API Definition

Table 5-18 POST [http://\[address\]\[:port\]/xxx/agv/agvCallbackService/agvCallback](http://[address][:port]/xxx/agv/agvCallbackService/agvCallback)

<b>API Name</b>	agvCallback
<b>Function</b>	Send the task executing status to the third-party platform.
<b>Protocol</b>	REST
<b>Provider</b>	Third-party platform
<b>Caller</b>	RCS-2000

<b>Remarks</b>	The exhaustive request parameters are more than parameters listed in the following table, the third-party platform can choose required parameters according to the business.				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Req.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	cooX	String	8	Opt.	X-coordinate of location code, unit: mm. This field exists when the task has been completed.
	cooY	String	8	Opt.	Y-coordinate of location code, unit: mm. This field exists when the task has been completed.
	currentPositionCode	String	32	Req.	Current position ID
	data	String	2000	Opt.	Custom content
	mapCode	String	16	Opt.	Map ID
	mapDataCode	String	32	Opt.	Location code on map, and it is unique. This field exists when the task has been completed.
	method	String	16	Req.	Name to describe the task executing status: "start"-task started, "outbin"-executing, "end"-task completed, "cancel"-cancel task; it is provided by RCS-2000
	podCode	String	16	Opt.	Rack ID, this field exists when the rack is carried.
	podDir	String	4	Opt.	Rack directions: "180"-leftward, "0"-rightward, "90"-upward, "-90"-downward; this field exists when the task has been completed.
	robotCode	String	16	Req.	AMR ID
	taskCode	String	64	Req.	Current task ID (UUID)

	wbCode	String	32	Opt.	Workstation ID, which consists of letters and digits, and it must be same with that configured by RCS-2000. This field exists when the task has been completed, and it is same with parameter <b>wbCode</b> of API <i>genAgvSchedulingTask</i> .
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message
	data	String	2000	Opt.	Custom content to be returned, such as task ID.
<b>Sample Codes</b>	<b>Request</b>	<pre>{   "reqCode": "1541954B96B1112",   "reqTime": "2019-04-03 10:08:06",   "cooX": "3000",   "cooY": "21999",   "currentPositionCode": "p02",   "mapCode": "AA",   "mapDataCode": "002069AA015172",   "method": "end",   "podCode": "100001",   "robotCode": "6001",   "taskCode": "test169E0F39740116Q",   "wbCode": "p02" }</pre>			
	<b>Response</b>	<pre>{   "code": "0",   "message": "successful",   "reqCode": "1541954B96B1112",   "data":"" }</pre>			

### 5.3.2 warnCallback

RCS-2000 sends the severe alarm which causes AMR stopping to the third-party platform. The alarm sending frequency is 10 seconds per time.



## API Definition

Table 5-19 POST http://[address][:port]/service/rest/agvCallbackService/warnCallback

<b>API Name</b>	warnCallback				
<b>Function</b>	RCS-2000 sends the severe alarm which causes AMR stopping to the third-party platform.				
<b>Protocol</b>	REST				
<b>Provider</b>	Third-party platform				
<b>Caller</b>	RCS-2000				
<b>Remarks</b>	The request URL should be "http://[address][:port]/service/rest/agvCallbackService/warnCallback", of which the "http://[address][:port]/service/rest" can be configured in the system parameters configuration of RCS-2000, and the corresponding configuration numbers are: 10012, 10013, and 10014.				
<b>Request</b>	<b>Parameter</b>	<b>Data Type</b>	<b>Max. Length</b>	<b>Req. or Opt.</b>	<b>Description</b>
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	data	Object			
	robotCode	String	16	Req.	AMR ID
	beginTime	String	64	Req.	Alarm start time
	warnContent	String	64	Req.	Alarm content, see <b>AMR Status</b> (Appendix B) for details.
	taskCode	String	64	Opt.	Task No.
<b>Response</b>	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.
	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message.

Sample Codes	Request	<pre>{   "reqCode": "1541954B96B1112",   "data": [{     "robotCode": "1001",     "beginTime": "2020-04-02 23:12:12",     "warnContent": "Platform disconnected",     "taskCode": "C002WWQQR"   },   {     "robotCode": "1002",     "beginTime": "2020-04-02 23:12:12",     "warnContent": "Guidance alarm",     "taskCode": "C002WWQQR33"   } ]</pre>
	Response	<pre>{   "code": "0",   "message": "successful",   "reqCode": "1541954B96B1112" }</pre>

### 5.3.3 bindNotify

Notify the third-party platform of the binding or unbinding operation between racks and locations, material batches and racks, or containers and bins.

#### API Definition

Table 5-20 POST `http://[address][:port]/service/rest/bindNotify`

API Name	bindNotify
Function	Notify the third-party platform of the binding or unbinding operation between racks and locations, material batches and racks, or containers and bins.
Protocol	REST
Provider	Third-party platform
Caller	RCS-2000
Remarks	<ul style="list-style-type: none"><li>The request URL should be "http://[address][:port]/service/rest/bindNotify", of which the "http://[address][:port]/service/rest" can be configured in the system parameters configuration of RCS-2000, and the corresponding configuration numbers are: 10012, 10013, and 10014.</li><li>You can specify the binding or unbinding type via the system configuration number 10026.</li></ul>

Request	Parameter	Data Type	Max. Length	Req. or Opt.	Description
	reqCode	String	32	Req.	Request ID, if a request is repeatedly submitted, the request ID must be same.
	reqTime	String	20	Opt.	Request timestamp, format: YYYY-MM-DD hh:mm:ss
	clientCode	String	16	Opt.	Third-party platform ID, e.g., PDA, HCWMS
	tokenCode	String	64	Opt.	Token ID, which is provided by RCS-2000 for third-party platform.
	method	String	16	Req.	Method name: "bindPodAndBerth" (bind and unbind rack with location), "bindPodAndMat" (bind and unbind material batch and rack), "bindCtnrAndBin" (bind and unbind container and bin).
	indBind	String	2	Req.	Bind or unbind: "1" (bind), 0 (unbind).
	bindParam	Object[list] Up to 2000 characters are allowed.			
	podCode	String	32	Opt.	Rack ID. It is valid when the value of <b>method</b> is "bindPodAndBerth" or "bindPodAndMat".
	berthCode	String	32	Opt.	Position ID, which is predefined with detailed coordinate information on map. It is valid when the value of <b>method</b> is "bindPodAndBerth".
	materialLot	String	32	Opt.	Material batch ID. It is valid when the value of <b>method</b> is "bindPodAndMat".
	ctnrCode	String	32	Opt.	Container ID. It is valid when the value of <b>method</b> is "bindCtnrAndBin".
	ctnrType	String	32	Opt.	Container type. It is valid when the value of <b>method</b> is "bindCtnrAndBin".
	stgBinCode	String	32	Opt.	Bin ID. It is valid when the value of <b>method</b> is "bindCtnrAndBin".
Response	code	String	6	Req.	Status code, see <b>Status Code</b> (Appendix A) for details.

## RCS-2000 API

	message	String	64	Req.	Returned status description, e.g., "successful".
	reqCode	String	64	Req.	Request ID, which is the same with that in corresponding request message.
<b>Sample Codes</b>	<b>Request</b>	<pre>{   "reqCode": "1541954B96B1112",   "reqTime": "2020-09-17 13:48:58",   "method": "bindPodAndBerth",   "indBind": "0",   "bindParam": [{     "berthCode": "AB1",     "podCode": "111111"   }] }</pre>			
	<b>Response</b>	<pre>{   "code": "0",   "message": "successful",   "reqCode": "1541954B96B1112" }</pre>			

## Appendix A. Status Code

The status code returned in the response message are defined in the table below.

### Status Code Description

code	Description
0	Succeeded.
1	Incorrect parameters.
6	No need to resend (the task of the same <b>reqCode</b> is not completed).
99	Unknown error, try again.
100	The task does not exist.

## Appendix B. AMR Status

The AMR common statuses are list in the table below.

### AMR Common Status

status	Description
1	Task completed.
2	Executing task
3	Abnormal task
4	Idle task
5	Robot stopped.
6	Lifting the rack.
7	Charging status
8	Curve movement
9	Full charge maintenance
11	Rack not recognized
12	Rack angle deflected
13	Motion library exception
14	Rack code unrecognized
15	Rack code mismatch
16	Lifting exception
17	Charging station exception
18	Battery not charging
20	Charging direction error
21	Platform command error
23	Abnormal unloading
24	The rack position deviated.
25	Robot not in the block zone
26	Retry putting down failed
27	Incorrect rack location

status	Description
28	Low battery for lifting
29	Robot reversing angle deflected
30	Lifting without rack
31	Blocking zone failed.
33	Rotation request temporarily failed.
34	Map switching code unrecognized



See Far, Go Further