Alibaba Cloud Application Configuration Management

Use Cases

Issue: 20190806

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. **You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.**
2. **No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company, or individual in any form or by any means without the prior written consent of Alibaba Cloud.**
3. **The content of this document may be changed due to product version upgrades , adjustments, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through Alibaba Cloud-authorized channels. You shall pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.**
4. **This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides the document in the context that Alibaba Cloud products and services are provided on an "as is", "with all faults " and "as available" basis. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity , applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not bear any liability for any errors or financial losses incurred by any organizations, companies, or individuals arising from their download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, bear responsibility for any indirect, consequential, exemplary, incidental, special, or punitive damages, including lost profits arising from the use**

or trust in this document, even if Alibaba Cloud has been notified of the possibility of such a loss.

5. By law, all the content of the Alibaba Cloud website, including but not limited to works, products, images, archives, information, materials, website architecture, website graphic layout, and webpage design, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of the Alibaba Cloud website, product programs, or content shall be used, modified , reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates . The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates).
6. Please contact Alibaba Cloud directly if you discover any errors in this document.

Generic conventions

Table -1: Style conventions

Style	Description	Example
	This warning information indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	 Danger: Resetting will result in the loss of user configuration data.
	This warning information indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	 Warning: Restarting will cause business interruption. About 10 minutes are required to restore business.
	This indicates warning information, supplementary instructions, and other content that the user must understand.	 Notice: Take the necessary precautions to save exported data containing sensitive information.
	This indicates supplemental instructions, best practices, tips, and other content that is good to know for the user.	 Note: You can use Ctrl + A to select all files.
>	Multi-level menu cascade.	Settings > Network > Set network type
Bold	It is used for buttons, menus , page names, and other UI elements.	Click OK.
`Courier font`	It is used for commands.	Run the `cd / d C :/ windows` command to enter the Windows system folder.
Italics	It is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	It indicates that it is a optional value, and only one item can be selected.	`ipconfig` *[-all|-t]*

Style	Description	Example
{} or {a|b}	**It indicates that it is a required value, and only one item can be selected.**	`swich {stand | slave}`

Contents

Legal disclaimer..... I
Generic conventions..... I
1 RocketMQ client traffic control design.....1
2 Build standard PaaS service configuration from the configuration center.....7
3 Simplify Spring Cloud micro-services environment configuration management with ACM..... 14

1 RocketMQ client traffic control design

RocketMQ is a commonly-used asynchronous RPC technology. This topic takes RocketMQ as an example to explain how to use ACM to implement traffic control over RocketMQ.

Brief introduction to RocketMQ

For RocketMQ calling, a typical traffic control method is to control traffic at the subscription end. Two traffic control methods are supported:

- Concurrent traffic control over message subscribers
- Consumption delay traffic control over message subscribers

The basic principle for consumption delay traffic control over message subscribers is to control the consumption speed by adding a delay upon each consumption at the client end. Under this circumstance, the fastest theoretical concurrent consumption speed is:

```
MaxRate = 1 / ConsumInte rval * Concurrent ThreadNumb er
```

For example, if the concurrent thread number (ConcurrentThreadNumber) is 20, and the consumption delay (ConsumInterval) is 100 ms, then according to the preceding formula:

```
200 = 1 / 0 . 1 * 20
```

Theoretically, we can limit concurrent consumption traffic within 200.

In comparison with the concurrent thread number traffic control, consumption delay traffic control has some advantages, such as that it’s easier to implement, it’s less dependent on RocketMQ client packages, and it doesn’t need the client to provide the dynamic adjustment API that controls the concurrent thread number.

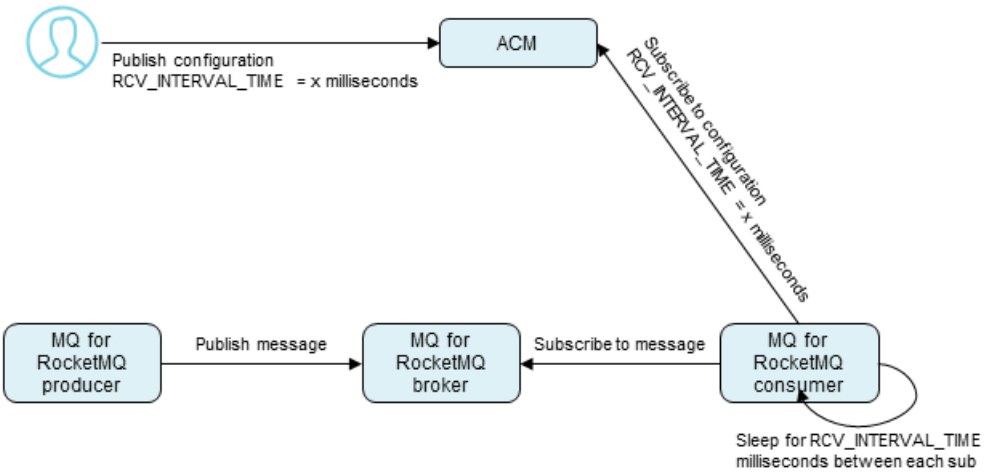
When using the above traffic control methods, if you want to implement dynamic global control under a distributed architecture, you can simply distribute traffic control parameters from the configuration center.

The following part elaborates how to implement dynamic global traffic control for asynchronous message consumption from the configuration center. configuration center. The example involves Alibaba Cloud’s RocketMQ and ACM (Application Configuration Management), and is based on Java.

> **Note:**
> The reason why we take RocketMQ as an example is that RocketMQ Consumer Client SDK currently doesn't support dynamic adjustment of the existing concurrent thread number, and we can solve the problem of dynamic RocketMQ consumption traffic control by dynamically adjusting consumption delay with ACM.

Basic principles of consumption-delay-based traffic control

As shown in the following diagram, the administrator or application publishes the consumption interval configuration (RCV_INTERVAL_TIME) through the ACM console , which is subscribed to by all RocketMQ consumption applications. Theoretically, it takes no more than 1 second for this configuration to be published and distributed to all clients (depends on network latency).

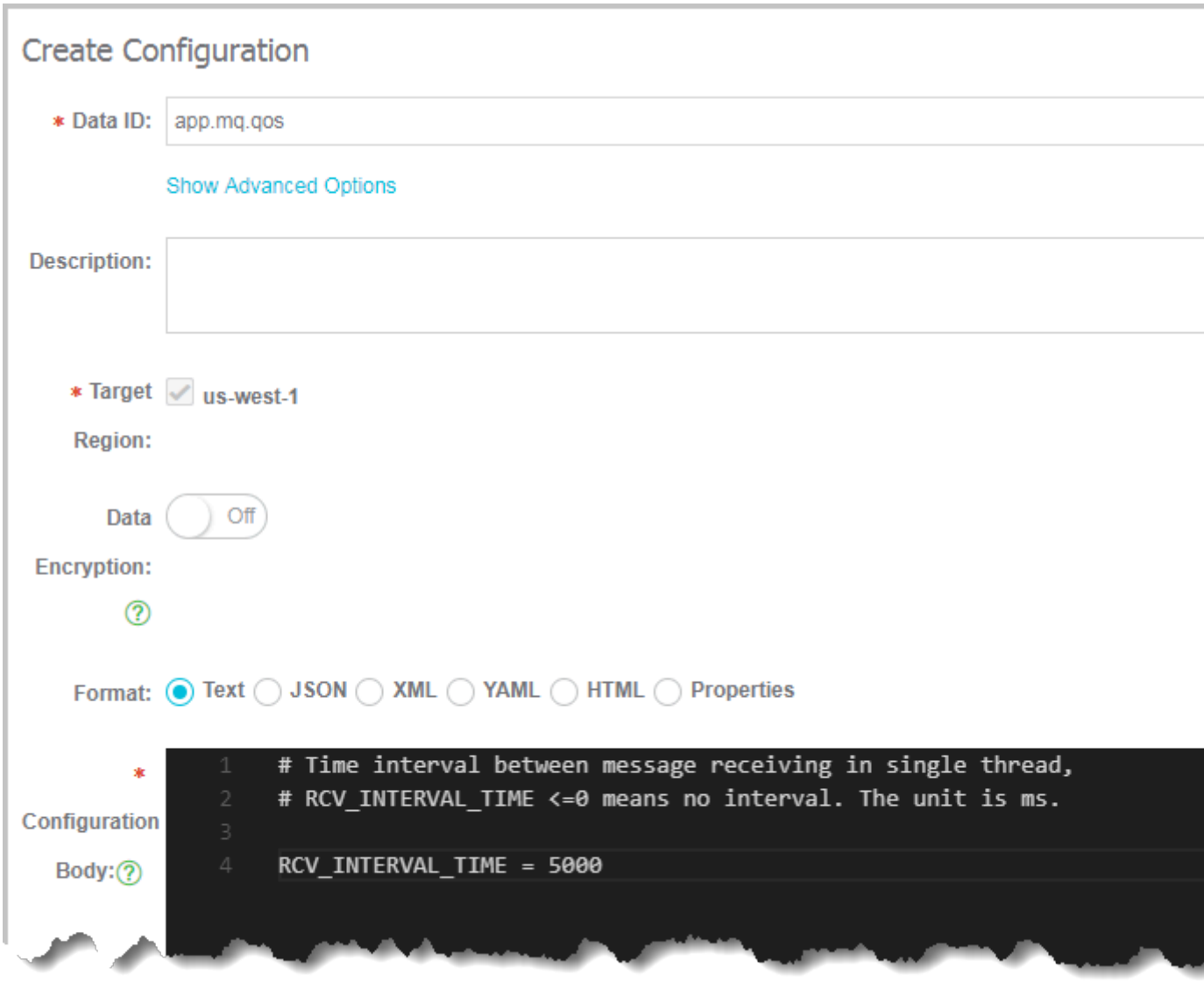


Sample code

This section provides the example code of dynamic global traffic control for asynchronous message consumption from the configuration center. For more information about SDK, see official documentation of RocketMQ and ACM.

Create ACM configuration

Create consumption delay parameters on ACM.



Set global consumption delay variable

1. Set global variable for consumption receipt delay.

```
// Initialize delay parameter for message receipt in
millisecon ds
 static int RCV_INTERV AL_TIME = 10000 ;
// Initialize configurat ion service , and then the
console will automatica lly retrieve the following
parameters with the example code
 ConfigServ ice . init (" acm . aliyun . com ", /* Tenant ID */"
xxx ", /* AK */" xxx ", /* SK */" yyy ");
// Actively retrieve configurat ion
 String content = ConfigServ ice . getConfig (" app . mq . qos
", " DEFAULT_GR OUP ", 6000 );
 Properties p = new Properties ();
 try {
     p . load ( new StringRead er ( content ));
     RCV_INTERV AL_TIME = Integer . valueOf ( p . getPropert y
(" RCV_INTERV AL_TIME "));
} catch ( IOExceptio n e ) {
     e . printStack Trace ();
```

```
}
```

2. Set ACM listener, and ensure the RCV_INTERVAL_TIME parameter is promptly updated upon modification of the configuration.

```
// Add a listener to the configurat ion during
initializa tion , which will send a callback notice
upon configurat ion change .
 ConfigServ ice . addListene r (" app . mq . qos ", " DEFAULT_GR
OUP ", new ConfigChan geListener () {
     public void receiveCon figInfo ( String configInfo ) {
         Properties p = new Properties ();
         try {
             p . load ( new StringRead er ( configInfo ));
             RCV_INTERV AL_TIME = Integer . valueOf ( p .
getPropert y (" RCV_INTERV AL_TIME "));
         } catch ( IOExceptio n e ) {
             e . printStack Trace ();
         }
     }
});
```

Set RocketMQ consumption delay logic

The complete example is as follows:

Note:

- In this example, the access of the RCV_INTERVAL_TIME parameter is intentionally kept unlocked, and the reason will not be elaborated here.
- Aliyun ONS Client doesn’t provide a dynamic concurrent thread number. The default concurrency is 20 by default. Therefore, in this example, we will use the consumption delay parameter to dynamically adjust QoS.

```
// The following code can be directly pasted into the
   Main () parameter
Properties properties = new Properties ();
properties . put ( PropertyKe yConst . ConsumerId , " CID_consum
er_group ");
properties . put ( PropertyKe yConst . AccessKey ," xxx ");
properties . put ( PropertyKe yConst . SecretKey , " yyy ");
properties . setPropert y ( PropertyKe yConst . SendMsgTim
eoutMillis , " 3000 ");
// Set TCP access domain name ( in this example , we
use a public cloud production environmen t ).
properties . put ( PropertyKe yConst . ONSAddr ,
 " http :// onsaddr - internet . aliyun . com / rocketmq /
nsaddr4cli ent - internet ");
Consumer consumer = ONSFactory . createCons umer ( properties );
consumer . subscribe (/* Topic */" topic - name ", /* Tag */ null ,
new MessageLis tener ()
{
    public Action consume ( Message message , ConsumeCon text
   context ) {
        // RocketMQ Subscribe QoS logical start ,
```

```
        //  Each   consuming   process   will   sleep   for
 RCV_INTERV  AL_TIME   seconds   with   100   ms   sleeping   cycle .
        //  Within   each   cycle ,  the   thread   will   check
 RCV_INTERV  AL_TIME   in   case   it ' s   set   to   a   smaller
 value .
        //  RCV_INTERV  AL_TIME  & lt ;=  0   means   no   sleeping .
         int   rcvInterva  lTimeLeft  =  RCV_INTERV  AL_TIME ;
         While  ( rcvinterva  ltimeleft >  0  ){
             if  ( rcvInterva  lTimeLeft  >  RCV_INTERV  AL_TIME ) {
                 rcvInterva  lTimeLeft  =  RCV_INTERV  AL_TIME ;
            }
             try  {
                 if  ( rcvInterva  lTimeLeft  >=  100 ) {
                     rcvInterva  lTimeLeft  -=  100 ;
                     Thread . sleep ( 100 );
                }  else  {
                     Thread . sleep ( rcvInterva  lTimeLeft );
                     rcvInterva  lTimeLeft  =  0 ;
                 }
            }  catch  ( Interrupte  dException   e ) {
                  e . printStack  Trace ();
            }
        }
        //  RocketMQ    subscribe   interval   logical   ends
         System . out . println (" Receive : " +  message );
        /*
         *  Put   your   business   logic   here .
         */
         doSomethin  g ();
         return   Action . CommitMess  age ;
    }
});
 consumer . start ();
```

Running result

Run the standalone Consumer for consumption. Assuming the message queue always has more messages than you can consume, the testing will be conducted in three stages, each lasting for five minutes. We can achieve the following results by using ACM configuration push service.

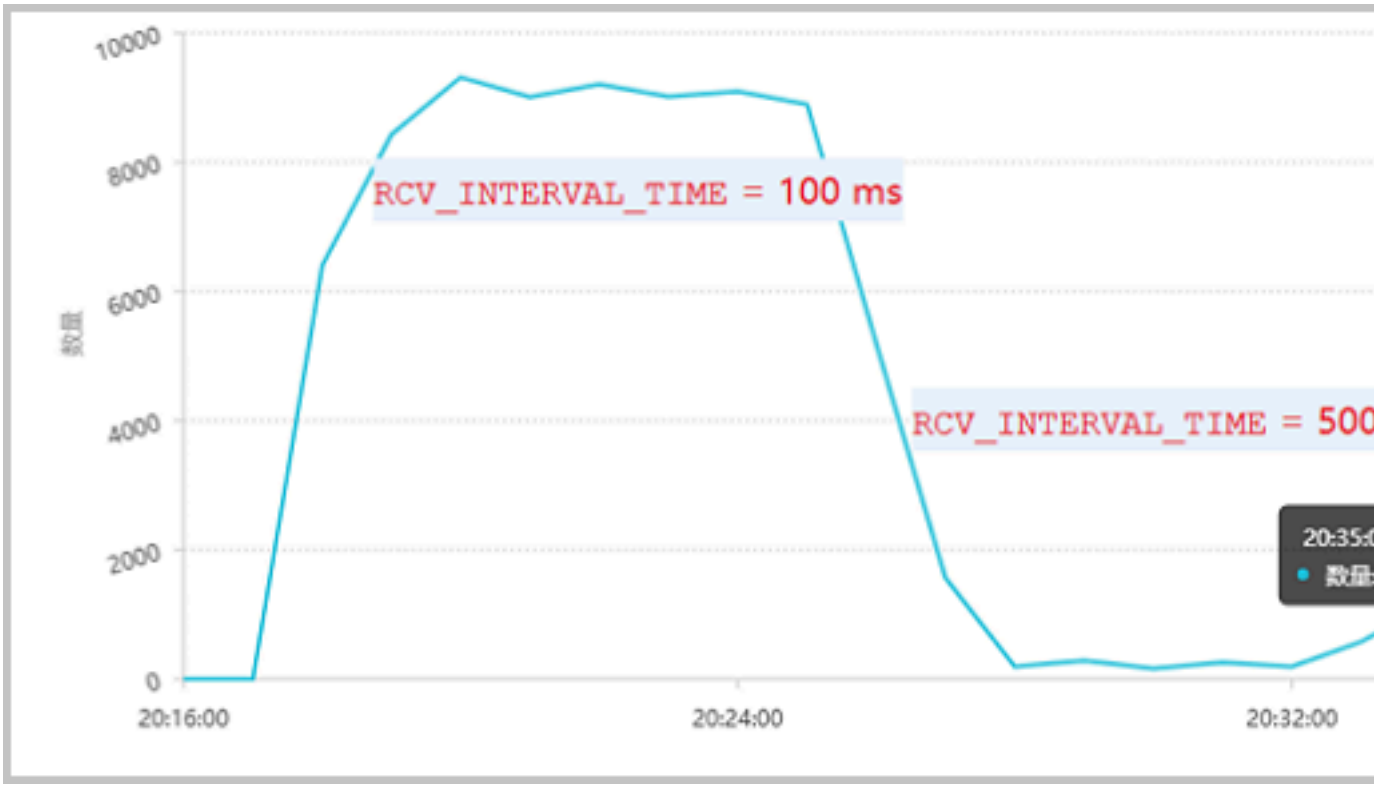
- RCV_INTERVAL_TIME = 100 ms
- RCV_INTERVAL_TIME = 5000 ms
- RCV_INTERVAL_TIME = 1000 ms

For a testing case with a standalone RocketMQ consumption business processing time of approximately 100 ms and a standalone concurrent thread number of 20, the testing results are as follows:

- RCV_INTERVAL_TIME = 100 ms: The average consumption performance is approximately 9000 tpm
- RCV_INTERVAL_TIME = 5000 ms: The average consumption performance is restricted to approximately 200 tpm

- RCV_INTERVAL_TIME = 1000 ms: The average consumption performance rises to approximately 1100 tpm

As expected, these results indicate that the consumption is inversely proportional to tpm. Most importantly, the application is not interrupted during the whole process. The traffic control push takes effect on distributed clusters in less than one second. The standalone performance result is as follows:



2 Build standard PaaS service configuration from the configuration center

MQ (Message Queue) is a common asynchronous RPC technology. This article explains how to conduct traffic control settings with standard configuration naming format, by taking traffic control over MQ as an example.

How configuration convention issues arise

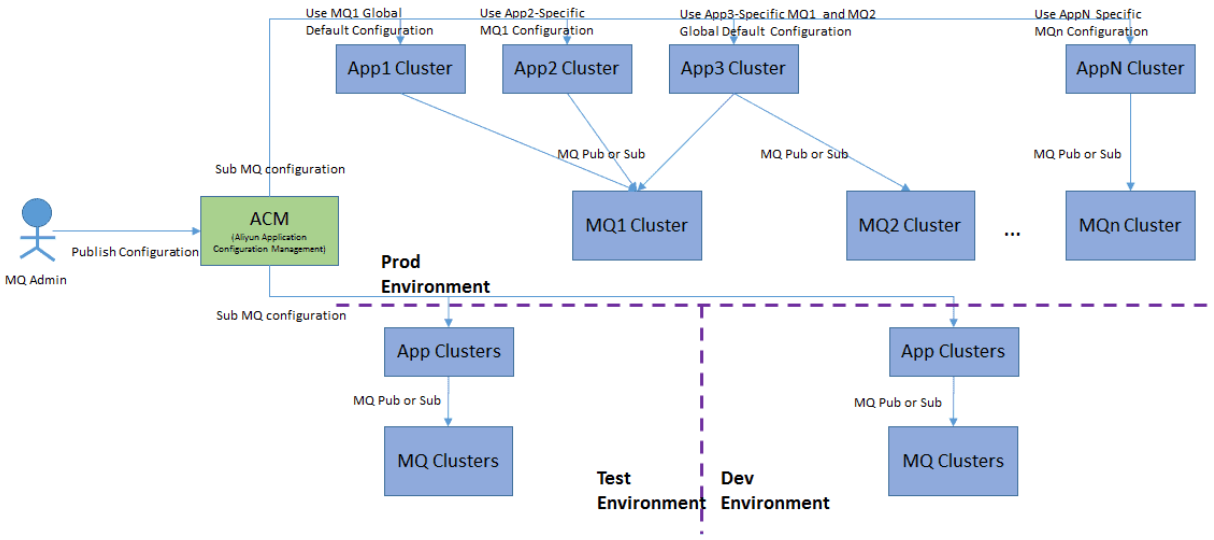
For single attribute configuration of a single application, you can directly use the following configuration file, without any configuration convention issues.

```
// Configurat  ion   directory   structure
-- app
   |-- src
      |-- config
         |-- applicatio  n . properties
//  Configurat  ion   content
 RCV_INTERV  AL_TIME = 20
```

However, when writing distributed rules for distributed PaaS services, the PaaS service provider (rather than the application party) often encounters many problems when designing the configuration. Some of the issues in the MQ traffic control scenario are:

- How to distinguish global configuration from local application configuration: For example, how can a PaaS service provider conduct global rule configuration and application-specific configuration when managing services provided by the platform.
- How to distinguish MQ services of different clusters: For example, how to distinguish configuration of MQ1 Cluster from MQ2 Cluster when the configuration naming is consistent?
- How to use one configuration center to isolate different environments such as Dev, Test, Staging, and Prod?

The preceding MQ traffic control scenario is illustrated as follows:



Obviously, improper configuration naming rules will affect the above configuration's ease of use.

The following is a detailed description of how to set the flow limit in a standardized configuration naming format. Before talking about the configuration naming conventions, it's necessary to understand the configuration structure organizing capability of the configuration center.

Configuration structure functions of the configuration center

In addition to centralized configuration management and subscription push, the configuration center also has the configuration structuring capability to help the administrator greatly simplify configuration management in various complex application scenarios.

1. Description of the configuration center's configuration structuring capability

- Tenant isolation: The configuration center has the capability to isolate configuration based on users and scenarios. With tenant isolation, different configurations in different tenants can have the same name but different authentication mechanisms.
- Minimum configuration set: Several configurations combine a configuration collection by configuring a central group. Use the minimum configuration collection to change and publish different configurations for unified publishing and processing. Configuration path, similar to file path or network domain name, decides the hierarchical relationship between different configuration sets.
- Key-Value form of a specific configuration: Users can set specific configuration content in the configuration center.

2. Comparison among different configuration centers with configuration structuring capabilities

To give you a more straightforward impression, we compared several configuration center products:

- **Alibaba Cloud ACM: Alibaba Cloud Application Configuration Management, formerly known as Diamond, is the earliest configuration center product in China. Alibaba Cloud ACM currently has different open-source versions on Git. It also has an enterprise version available from Alibaba Cloud.**
- **Spring Cloud Config: Spring Cloud official configuration center tool, mainly used in the Java Spring industry.**
- **ZooKeeper: With partial configuration center capabilities, ZooKeeper is positioned in distributed coordination information management, and can only be used as a configuration center for a moderate application scale. Considering its wide application scope, we included it in the comparison.**

Comparison details are as follows:

Function	ACM	Spring Cloud Config	ZooKeeper
Tenant-based isolation	Namespace and Group isolation. Different Namespaces require different AK/SK authentications, which are not required for Group.	A Git project is a tenant.	No readily available tenant isolation technology.

Function	ACM	Spring Cloud Config	ZooKeeper
Minimal configuration set	The configuration set marked with DataID is the minimal configuration set. The configuration set doesn't have the concept of path, but can be queried by using wildcard characters if the configuration set is properly designed. In this way, the similar effect of a configuration path can be achieved.	A configuration file in a Git project is the minimal configuration set, and doesn't have the concept of configuration path.	Znode is the minimal configuration set, and has the concept of configuration path.
Key-Value configuration	KV content is assembled by the user at will under DataID in any formats, such as properties, Json, XML, and so on. Validation function is provided on the management page.	Set KV by using the properties configuration file.	Set KV by setting the Value of Znode, and the content format is not restricted.

To sum up, ACM has a relatively higher degree of flexibility in both tenant isolation and minimal configuration set. The next part covers how to use ACM's Namespace, Group, DataID and other configuration functions to design a suitable configuration structure to implement QoS traffic control policies.

Best practices for distributed service configuration design based on the configuration center

1. Configuration structure

To meet functional needs of MQ configuration, we can use the following configuration methods in combination with the ACM features.

- Isolate MQ configurations for different environments with different namespaces. For example:
  - Use ProdEnv namespace for the production environment, TestEnv for the test environment, and DevENV for the development environment.
  - Different environments are automatically isolated with AK/SK, which further enhances the security.
- MQ services provided by different clusters are distinguished with Group to isolate configurations and simplify the access methods.
  - For example, for MQ clusters that specifically serve the subordinate departments and core transaction departments, as well as MQ clusters that specifically serve the subordinate departments and transaction departments, we can use Group to distinguish different global configurations. By doing this, all applications use the same company's/subsidiary's AK/SK (or similar authentication system key) in the production system, which simplifies the deployment, effectively isolates the configurations of different clusters, and reduces the configuration complexity.

```
DataID :  mq . global . qos
Group :  Trading

DataID :  mq . global . qos
Group :  ProductCat  egory
```

- Global configurations are stored in the form of configuration items with unified DataIDs.
  - Wherein, the configuration ID begins with `mq` . `global` indicates global configuration, `qos` indicates qos configuration, and default value can be used for Group.

```
DataID :  mq . global . qos
Group :  Default_Gr  oup
```

- Application local configuration IDs are named with the same prefix qos.*
  - Configuration Ids begin with `mq` . `app` .`[ appname ]` stands for the app configuration item to be reloaded, and default value can be used for Group.

```
DataID :  mq . app . app1 . qos
Group :  Default_Gr  oup
DataID :  mq . app . app2 . qos
```

```
Group : Default_Gr oup
```

3. Settings of KVs for specific configurations

For many configuration center products, such as Apollo, ACM System Manager Parameter Store, a specific configuration is the configuration center's smallest management unit. You need to set KVs of configurations one by one at a certain level. Two common practices are:

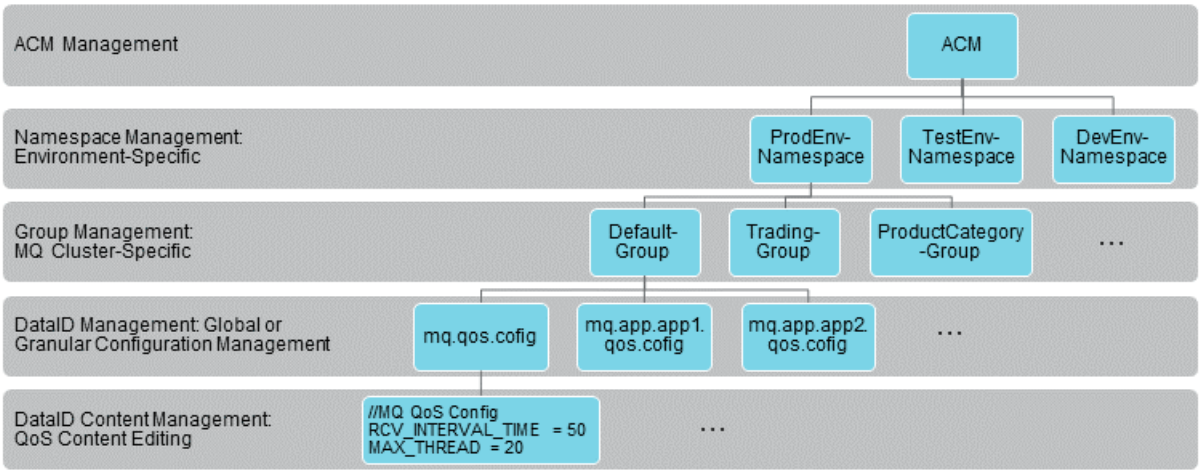
- Similar to the preceding configuration center, save each key to a unique DataID. For example:
  - Set mq.global.qos.RCV_INTERVAL_TIME to 50
  - Set mq.global.qos.MAX_THREAD to 20
- Combine common configurations into the same DataID, and save them in the same configuration file (supported formats are Properties, JSON, XML, and so on)
  - For example, setting of mq.global.qos is as follows:

```
// MQ traffic control QoS settings
 RCV_INTERV AL_TIME = 50
 MAX_THREAD = 20
```

In practice, the second method is found not only to have greater flexibility, multiple configurations are also supported for simultaneous release in one change, which reduces performance overhead, in theory, it has also achieved the atomic action effect of changing mass change.

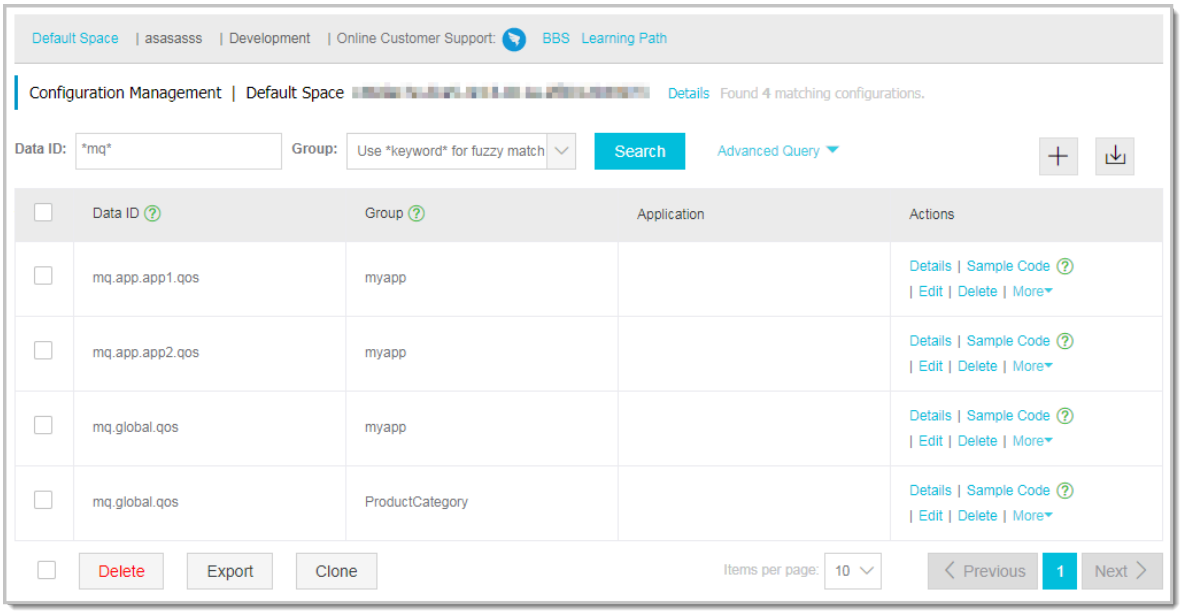
4. Configuration structure diagram

Configuration structure diagram of the preceding design is shown as follows:



5. Advantages of the scheme

- Isolating configurations for different environments with Namespaces allows MQ configuration items to use the same names in different Namespaces. Different namespaces are isolated by the administrator, program AK/SK, and permission settings, allowing you to centrally manage configuration items without affecting other environments.
- Isolating different clusters in the same environment by using Groups ensures the configuration consistency of different clusters (for example the configuration name doesn't change), simplifies code, and logically isolates different cluster configurations.
- Standard naming settings of the minimal configuration set DataIDs allow MQ clients to conveniently find both the MQ default global configurations, and their own application-specific configurations. In addition, on the Configurations page, administrators can add an asterisk (*) to each end of the keyword to easily filter out all MQ rules. For example:



References

- #unique_6

3 Simplify Spring Cloud micro-services environment configuration management with ACM

This article describes how to use Alibaba Cloud Configuration Center (ACM) in concert with Spring Cloud to help you simplify environment configuration management in a micro-service architecture, by testing different databases connected to the production environment and configuring different data source (including connection pool) parameters.

Environment attributes of configuration

During the continuous delivery of the system, the diversity and complexity of the system’ s final running environment undoubtedly make it harder for us to manage configurations. Eugen Paraschiv briefly discussed this in his post Configuration Must Be Environment Specific. It is also elaborated in depth in the “Containerization, scheduling, and configuration management” section in Configuration management challenges in modern application architectures.

Due to the differences between the configurations of different environments, the artifacts of those environments are not consistent, and sometimes Docker can't deliver the "build once, run anywhere" experience as expected. Here are some simple examples to help you better understand.

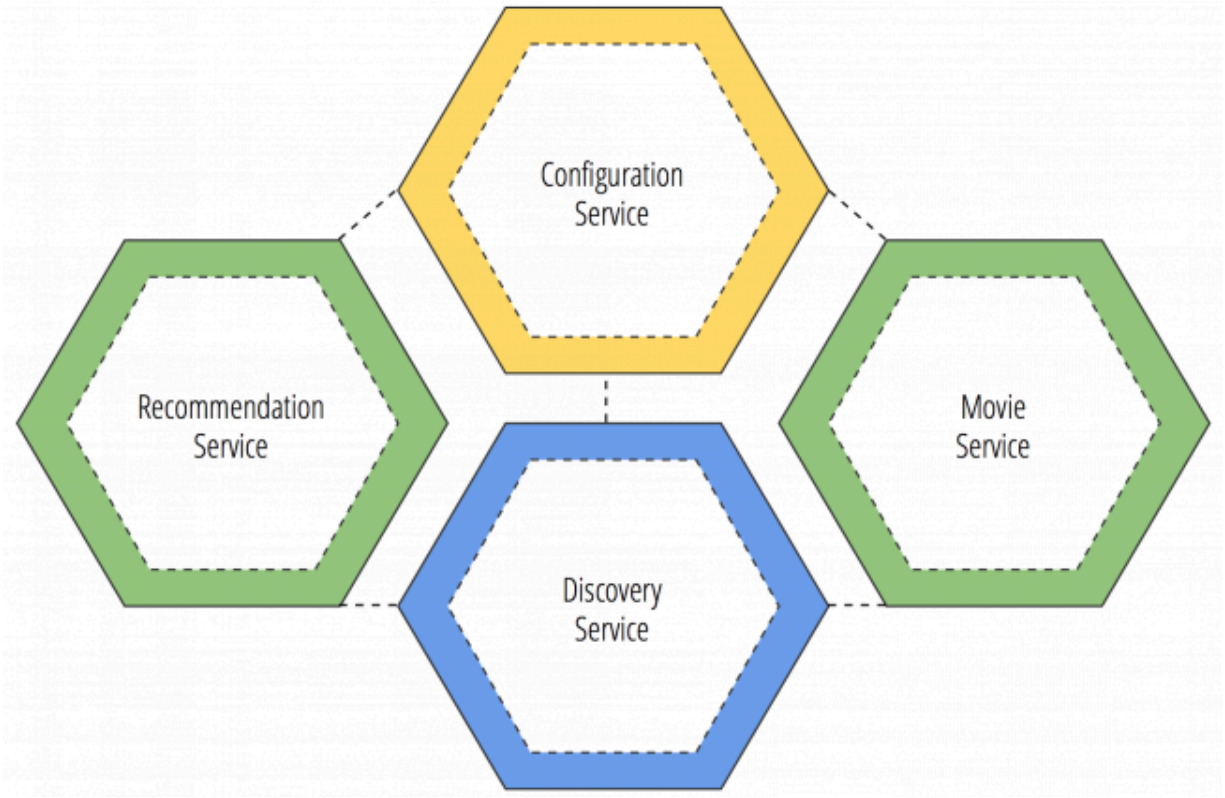
- The logLevel should be set to DEBUG in the development environment, INFO in the staging environment, and WARNING in the production environment.
- The database runs on a 4-core 8 GB-RAM machine in the development environment , but on a 32-core 96 GB-RAM machine in the production environment.
- The maximum thread number in the execution thread pool of the daily environment should be set to 15, while this number should be larger in the production environment, which is 150 by default.
- In the online environment, application data sources in the Central Data Center need to connect to Database A, while application data sources in Shenzhen Data Center should connect to Database B due to the proximity.
- Two-way synchronization switch should be switched off only in and only in Mini Taobao environment.

- The new features should be made available only in the online Hangzhou Unit environment rather than other unit environments.

In this article we will briefly describe how to use Alibaba Cloud ACM in Spring Cloud to replace Spring Cloud Config in simplifying environment configuration management, and help you understand the ACM-based solutions to simplify micro-service environment configuration management. We will also list the pros and cons of ACM and Spring Cloud Config.

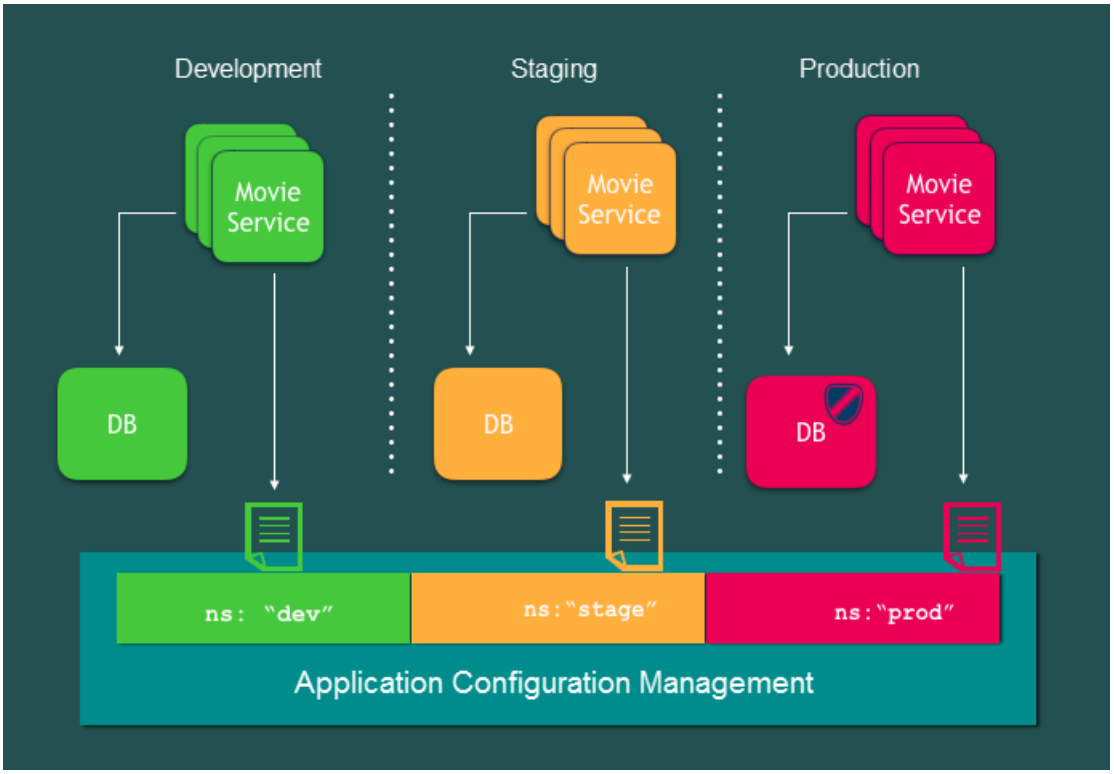
User stories

In daily engineering practice, we often picture a simple scenario with a user story to facilitate the elaboration and communication. This is an illustration used for preaching in the early stages:



Taking Movie Service as an example. Let's assume that we need to retrieve a list of all movies from the relational database MySQL(RDS). We only need the top-configuration machines for the production database, and we need to use different databases in the testing, pre-release, and production environments. Therefore, our applications need to have different data source configuration, connection pool configuration, database security configuration, and so on in different environments.

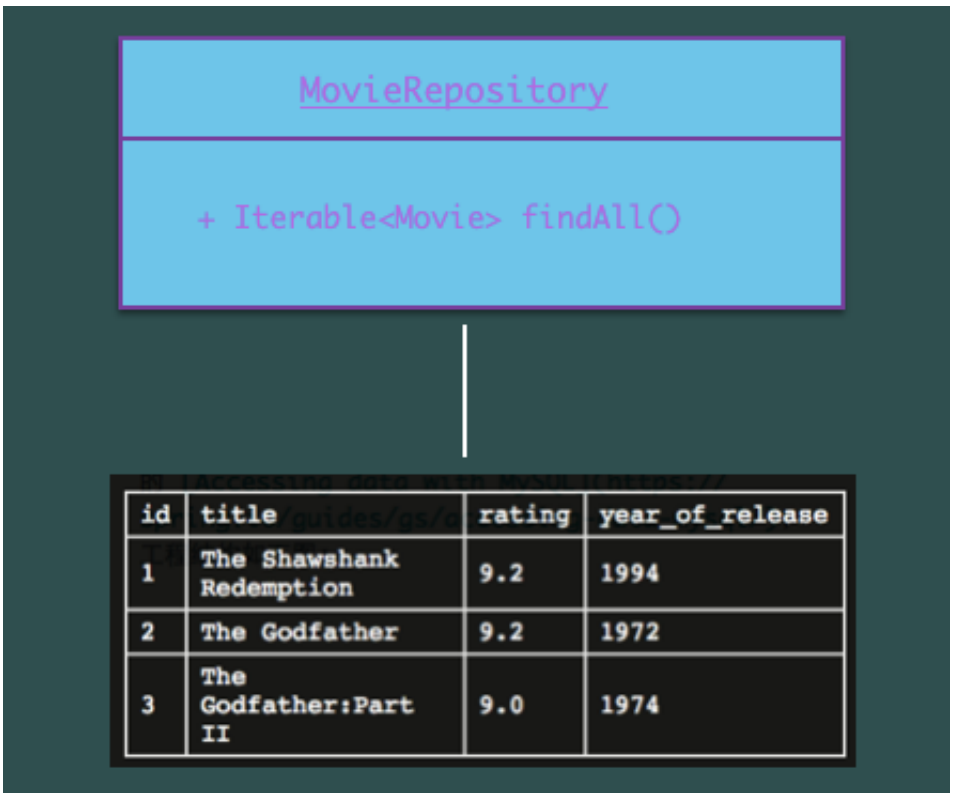
The following figure shows how to map different environments with ACM Namespace , and set different data source configurations for Movie Service in different running environments.



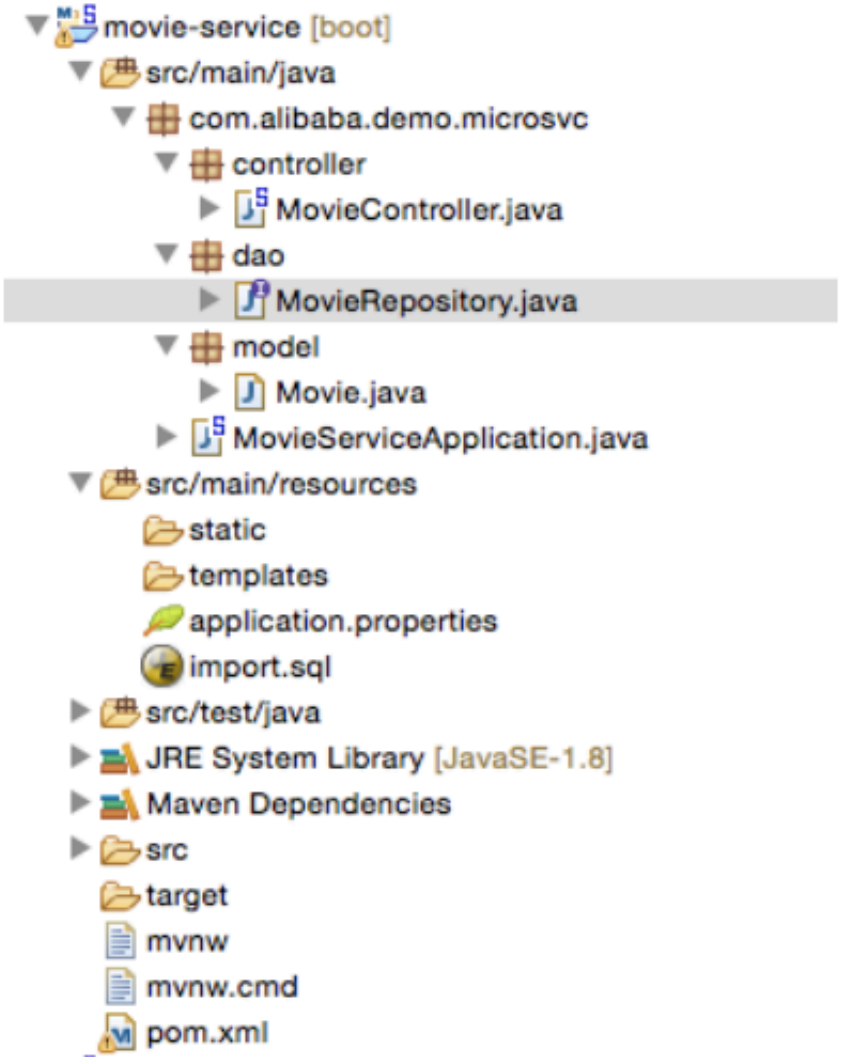
Create micro-service: movie service

- Create Spring Boot Starter micro-service: movie service

  Movie service's business logic is very straightforward: to list all movies in MySQL (RDS):



  Here we created a standard JPA application (similar to the sample project on Spring official website Accessing data with MySQL). The project structure is as shown in the following figure:



- **Introduce JPA, MySQL, connection pool HikariCP and WEB dependencies**

```
< dependency >
   < groupId > org . springfram  ework . boot </ groupId >
   < artifactId > spring - boot - starter - web </ artifactId >
</ dependency >
< dependency >
   < groupId > org . springfram  ework . boot </ groupId >
   < artifactId > spring - boot - starter - data - jpa </ artifactId
 >
</ dependency >
< dependency >
   < groupId > mysql </ groupId >
   < artifactId > mysql - connector - java </ artifactId >
   < scope > runtime </ scope >
</ dependency >
< dependency >
   < groupId > com . zaxxer </ groupId >
   < artifactId > HikariCP </ artifactId >
   < version > 2 . 7 . 6 </ version >
```

```
</ dependency >
```

- Create MySQL(RDS) database and users

```
mysql > create database db_example ; -- Create the new
database
mysql > create user ' springuser '@' localhost ' identified
by ' ThePasswor d '; -- Creates the user
mysql > grant all on db_example .* to ' springuser '@'
localhost '; -- Gives all the privileges to the new
user on the newly created database
```

For detailed steps, see the "Create the database" section in Accessing data with MySQL.

- Create a web Controller

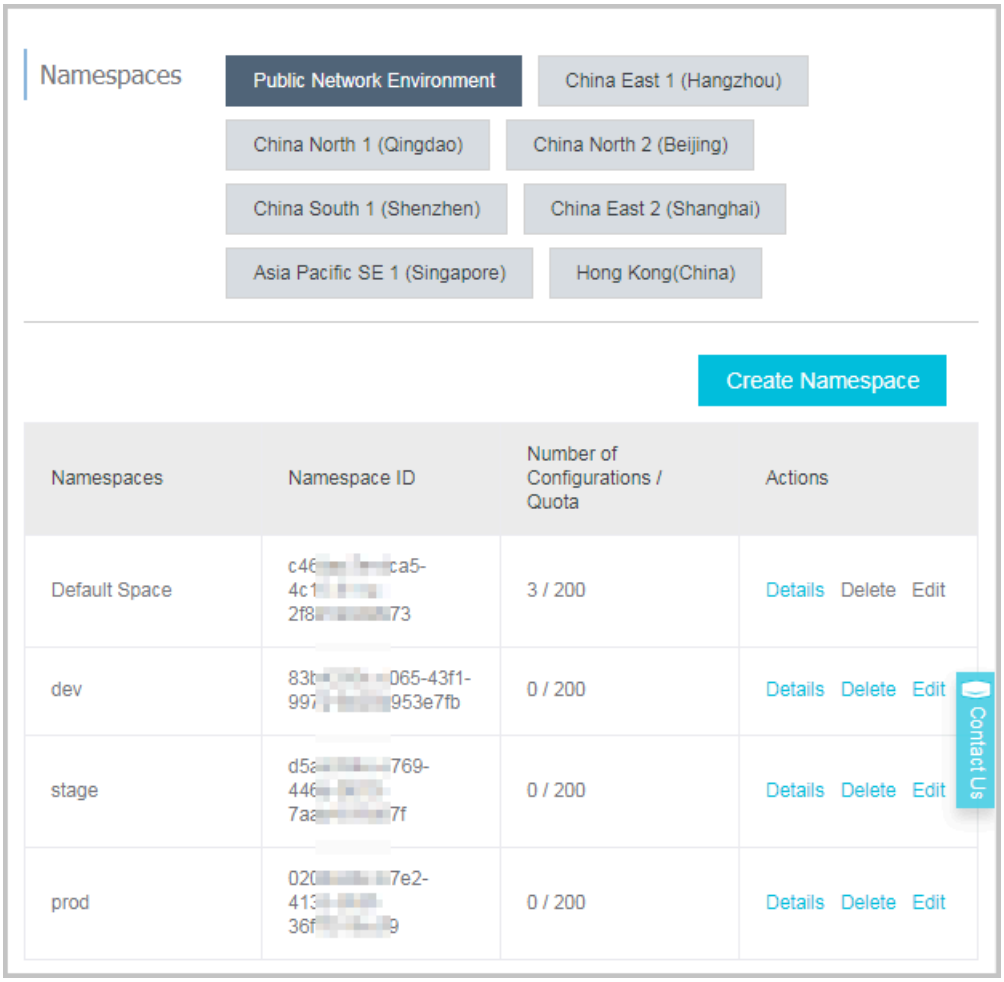
```
package com . alibaba . demo . microsvc . controller ;
 import org . springfram ework . beans . factory . annotation .
Autowired ;
 import org . springfram ework . web . bind . annotation .
RequestMap ping ;
 import org . springfram ework . web . bind . annotation .
ResponseBo dy ;
 import org . springfram ework . web . bind . annotation .
RestContro ller ;
 import com . alibaba . demo . microsvc . dao . MovieRepos itory
;
 import com . alibaba . demo . microsvc . model . Movie ;
@ RestContro ller
 public class MovieContr oller {
    @ Autowired
     MovieRepos itory movieRepos itory ;
    @ RequestMap ping ("/ list - movies ")
     public @ ResponseBo dy Iterable < Movie > listMovies () {
          return movieRepos itory . findAll ();
    }
}
```

Use Namespace in ACM to create isolated environment configuration

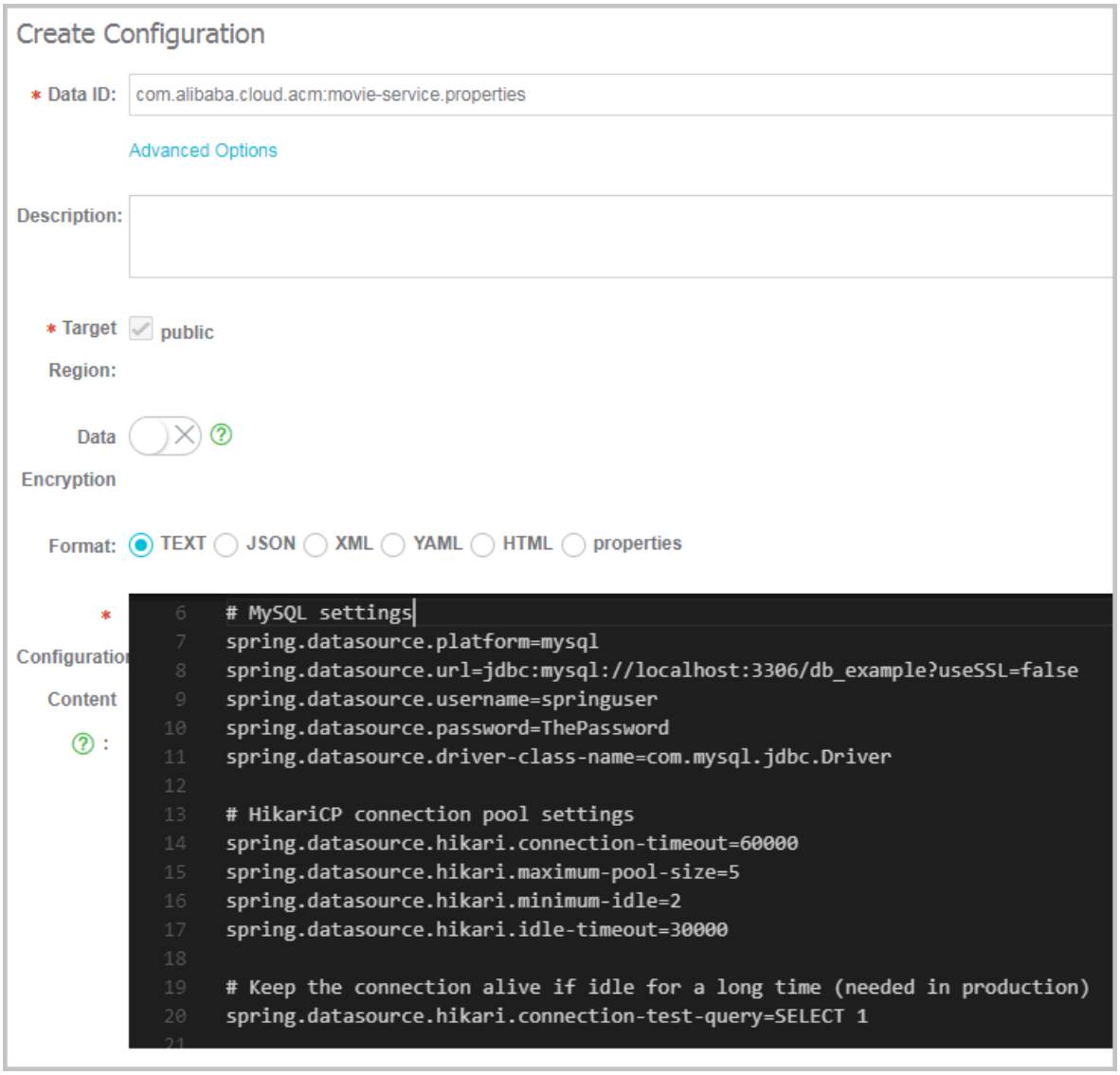
Note:

To use ACM on Alibaba Cloud, you must enable this service. For instructions, see Activate ACM. Once you activate the service and log on, you can create namespaces and configurations in the ACM console.

- **Create three environments in ACM: dev, stage, and prod**



- Create configuration respectively for dev, stage, and prod environments



In the previous step, we set different values for the same configuration item in different environments. Taking the `spring . datasource . url` configuration item as an example, we connect different databases to different environments by setting different URLs, and only enables SSL (useSSL=true) in the production environment.

```
dev :
    spring . datasource . url = jdbc : mysql :// localhost : 3306 /
db_example ?  useSSL = false
prod :
```

```
    spring . datasource . url = jdbc : mysql :// 30 . 5 . 101 . 169 :
3306 / db_example ?  useSSL = true
```

In addition, we have set a larger database connection pool and a smaller timeout value for the prod environment.

```
dev :
    spring . datasource . hikari . connection - timeout = 60000
    spring . datasource . hikari . maximum - pool - size = 10
prod :
    spring . datasource . hikari . connection - timeout = 15000
    spring . datasource . hikari . maximum - pool - size = 200
```

To make it easier to debug, we only enables SQL Trace in the dev environment.

```
dev :
    spring . jpa . show - sql = true
```

Integrate Movie Service with ACM

Next, we integrate Movie Service with ACM to obtain the corresponding environment configuration from ACM. For instructions on how to use ACM in Spring Cloud, see Spring Cloud ACM.

- Introduce ACM dependencies to Movie Service

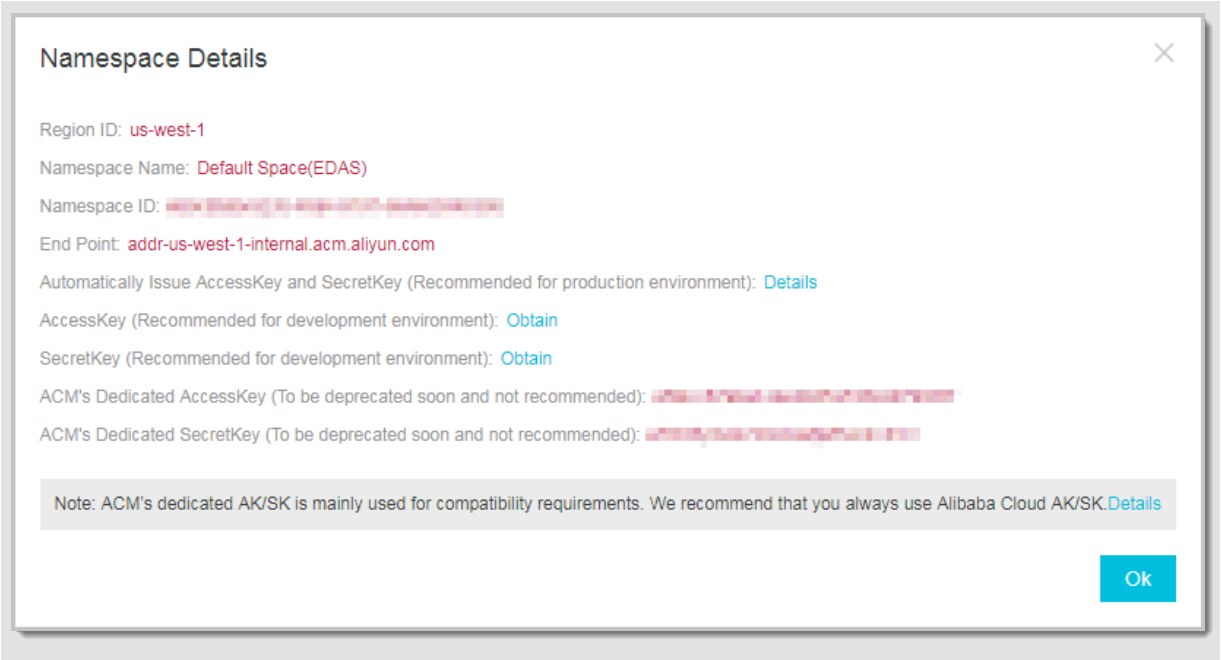
```
< dependency >
   < groupId > com . alibaba . cloud </ groupId >
   < artifactId > spring - cloud - starter - acm </ artifactId >
   < version > 1 . 0 . 1 </ version >
</ dependency >
```

- Configure ACM connection information, namespace, accessKey, secretKey, and other information in application.properties

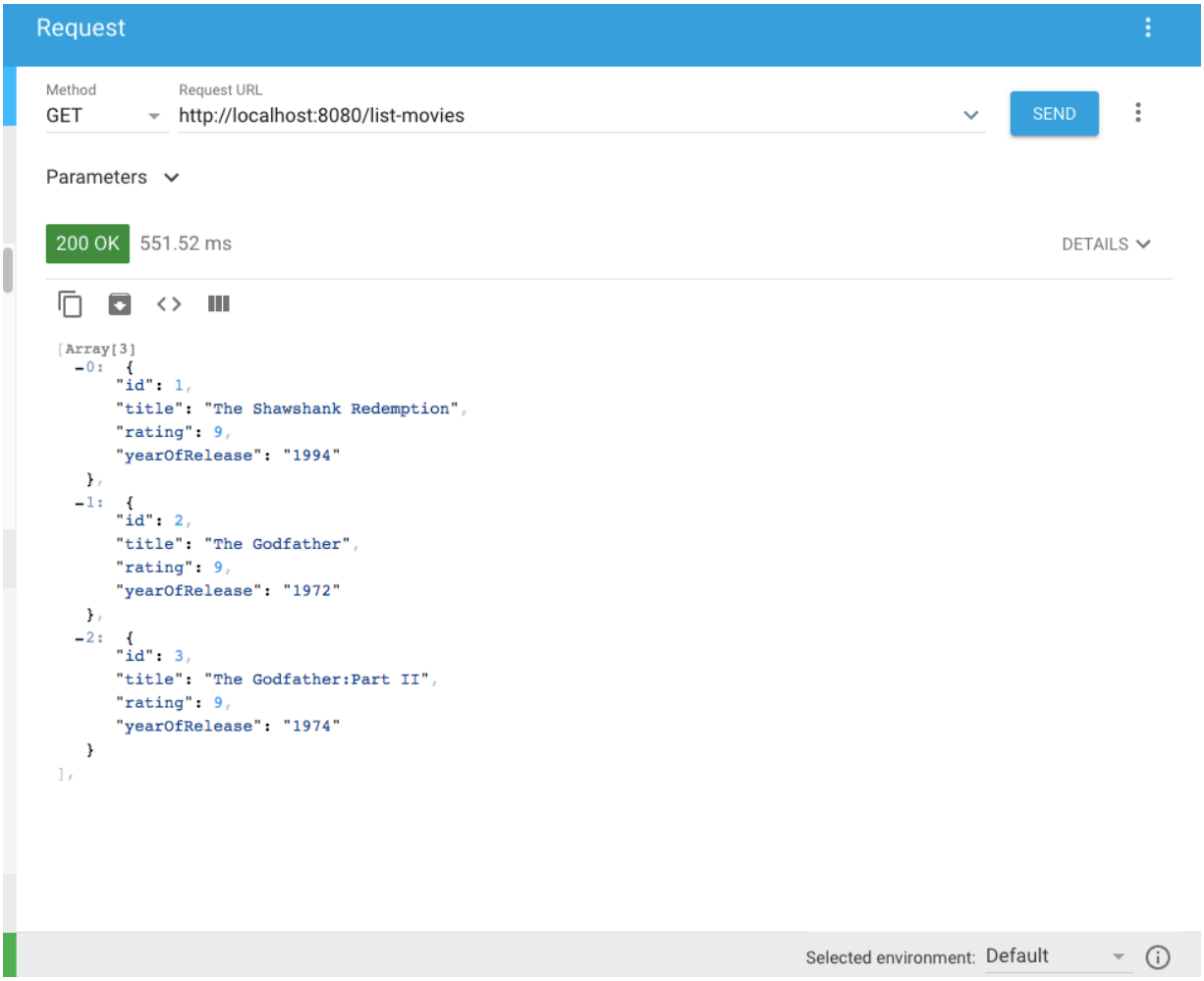
```
spring . applicatio  n . name = movie - service
spring . applicatio  n . group = com . alibaba . cloud . acm
alibaba . acm . endpoint = acm . aliyun . com
alibaba . acm . namespace =< your_names  pace_id >
alibaba . acm . accessKey =< your_ak >
alibaba . acm . secretKey =< your_sk >
```

Note:

You can find your namespace_id, accessKey, secretKey, and other information in “namespace details” or “code example of configuration”.

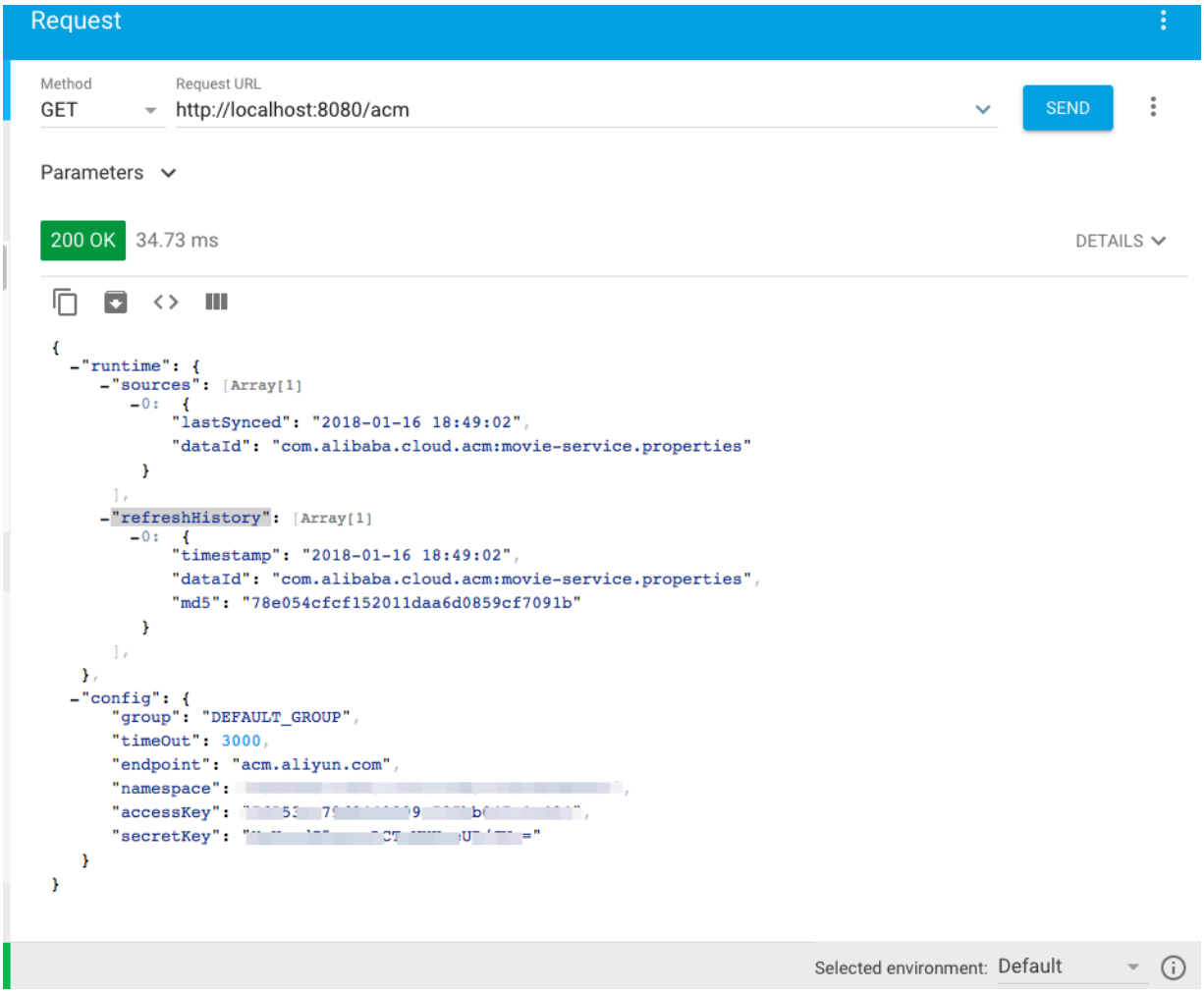


Access Movie Service from your browser



View pushing and refreshing history of ACM configurations

If spring-boot-starter-actuator dependency has been introduced to Movie Service, and `management.security.enabled=false` is set in application.properties, you can view the application's configuration consumption and refreshing history from endpoint `http://<<ip:port>>/acm`.



You can also view the configuration's push track, configuration version, and other information from the ACM console. For instructions, see ACM documentation.

Brief comparison between ACM and Spring Cloud Config

What's compared	Spring cloud config	Alibaba Cloud ACM
Spring Cloud seamless integration	Supported	Supported
Source code distribution method	Open source	To be open-sourced soon
Billing method	Free of charge	Free of charge

What’s compared	Spring cloud config	Alibaba Cloud ACM
Large scale (over 100,000 configuration items) production verification	No publicly available cases of large scale production verification	Verified with Alibaba data center’s production environment featuring millions of configuration items, with over 100 million configuration changes pushed every day, and the Double 11 shopping spree and many other demanding scenarios
Configuration management UI console	No console, dependent on IDE, GIT, and other third-party tools	Professional configuration management UI console
Multi-language support	Mainly supports Java ecosystem, without any native clients for other languages	Supports Node.js, C++ and other native multi-language clients
Multi-data center, Local active-active disaster recovery, multi-zone, and other architectures	Dependent on support of GIT, ZooKeeper, and so on, without an explicit official statement	Supported
Configuration change push	Dependent on RabbitMQ/ Kafka	Built-in push mechanism, without external dependency
Timeliness of large scale configuration push	Dependent on SLA and Webhooks such as GIT Webhooks and so on. Enterprise level large scale production capability is yet to be verified.	Industrial level production in milliseconds
Audit capability for configuration changes	Weak	Built-in audit mechanism (with an audit capability conforming to National Graded Protection of Information Security - Level 3)

What’s compared	Spring cloud config	Alibaba Cloud ACM
Push track	Unable to view real-time monitoring that is configured to push to the client	Provides configuration change push tracks for monitoring configuration change push status
Data isolation	Supports application, profile, label, git repo, and other isolation policies	In addition to isolation policies provided by Spring Cloud, ACM supports multi-tenant, app, data_id, group and other multi-level isolation policies
Production and O&M cost	High (must have sufficient GIT/RabbitMQ knowledge and talent reserve)	Low (no third-party component dependencies)
High availability	N/A (customers must assume all risks)	99.99% (Alibaba Cloud assumes the risks)
Secure communication	Supports SSL	Supports SSL
Disaster tolerance	Two levels (storage, server cache)	Three levels (plus local disaster recovery of the client)

Download project

Sample project used in this article can be downloaded from movie-service.tar.gz.

This project has passed the test in the following environments:

- Spring Cloud Edgware.RELEASE
- Spring Boot 1.5.9. RELEASE
- HikariCP 2.7.6
- MySQL 5.7.11
- ACM 4.2.0
- ACM Spring Cloud SDK 1.0.1

Note:

Before running this project locally, make sure to set your own ACM accessKey and secretKey in application.properties.