Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: December 14, 2018

C. Yang, Ed.
Y. Liu, Ed.
South China University of Technology
C. Chen
Inspur
G. Chen
GSTA
Y. Wei
Huawei

June 12, 2018

A Massive Data Migration Framework
draft-yangcan-ietf-data-migration-standards-00

Abstract

This document describes a standardized framework for implementing the massive data migration between traditional databases and big-data platforms on the cloud via Internet, especially for an instance of Hadoop data architecture. The main goal of the framework is to provide more concise and friendly interfaces for users more easily and quickly migrate the massive data from a relational database to a distributed platform for a variety of requirements, in order to make full use of distributed storage resource and distributed computing capability to solve the bottleneck problems of both storage and computing performance in traditional enterprise-level applications. This document covers the fundamental architecture, data elements specification, operations, and interface related to massive data migration.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 14, 2018.

Table of Contents

1. Introduction

With the widespread popularization of cloud computing and big data technology, the scale of data is increasing rapidly, and the distribution computing requirements are more significant than before. For a long time, a majority of companies have usually use relational databases to store and manage their data, a great amount of structured data exist still and accumulate with the business development in legacies. With the dairy growth of data size, the storage bottleneck and the performance degradation for the data when analyzing and processing have become pretty serious and need to be solved in globe enterprise-level applications. This distributed platform refers to a software platform that builds data storage, data analysis, and calculations on a cluster of multiple hosts. Its core architecture involves in distributed storage and distributed computing. In terms of storage, it is theoretically possible to expand capacity indefinitely, and storage can be dynamically expanded

horizontally with the increasing data. In terms of computing, some key computing frameworks as mapreduce can be used to perform parallel computing on large-scale datasets to improve the efficiency of massive data processing. Therefore, when the data size exceeds the storage capacity of a single-system or the computation exceeds the computing capacity of a stand-alone system, massive data can be migrated to a distributed platform. The ability of resource sharing and collaborative computing provided by a distributed platform can well solve large-scale data processing problems. The document focuses on putting forward a standard for implementing a big data migration framework through web access via Internet and considering how to help users more easily and quickly migrate the massive data from a traditional relational database to a cloud platform from multiple requirements. Using the distributed storage and distributed computing technologies highlighted by the cloud platform, on the one hand, it solves the storage bottleneck and the problem of low data analyzing and processing performance of relational databases. Based on the access by web, the framework supports open work state and promotes globe applications for data migration.

Note: It is also permissible to implement this framework in non-web.

2. Definitions and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

The following definitions are for terms used in the context of this document.

- "DMOW": Its full name is "Data Migration on Web",it means data migration based on web.

- "Cloud Computing": Cloud computing is a pay-per-use model that provides available, convenient, on-demand network access, the user enters a configurable computing resource sharing pool (resources include network, server, storage, application software, services), these resources can be provided quickly, with little administrative effort or little interaction with service providers.

- "Big Data": A collection of data that cannot be captured, managed, and processed using conventional software tools within a certain time frame. That is a massive, high growth rate and diversified information assets that require new processing modes to have stronger decision-making power, insight and process optimization capabilities.

- "Data Migration": The data migration described in this document is aimed at the data transfer process between a relational database and a cloud platform.

- "Data Storage": Data is recorded in a format on the computer’s internal or external storage media.

- "Data Cleansing": It is a process of re-examining and verifying data. The purpose is to remove duplicate information, correct existing errors, and provide data consistency.

- "Extraction-Transformation-Loading(ETL)": The processing of user database or data warehouse. That is, data is extracted from various data sources, converted into data that meets the needs of the business, and finally loaded into the database.

- "Distributed Platform": A software platform that builds data storage, data analysis, and calculations on a cluster of multiple hosts.

- "Distributed File System": The physical storage resources managed by the file system are not directly connected to the local node. Instead, they are distributed on a group of machine nodes connected by a high-speed internal network. These machine nodes together form a cluster.

- "Distributed Computing": A computer science discipline studies how to divide a problem that requires a very large amount of computing power into many small parts and it is coordinated by many independent computers to get the final result.

- "Apache Hadoop": An open source distributed system infrastructure that can be used to develop distributed programs for large data operations and storage.

- "Apache HBase": An open source, non-relational, distributed database. Used with the Hadoop framework.

- "Apache Hive": It is a data warehouse infrastructure built on Hadoop. It can be used for data extraction-transformation-loading(ETL), it is a mechanism that can store, query, and analyze large-scale data stored in Hadoop.

- "HDFS": A Hadoop distributed file system designed to run on general-purpose hardware.

- "MapReduce": A programming model for parallel computing of large-scale data sets (greater than 1 TB).

- "Spark": It is a fast and versatile computing engine designed for large-scale data processing.

- "MongoDB":It is a database based on distributed file storage designed to provide scalable, high-performance data storage solutions for web applications.

3. Specific Framework Implementation Standards

The main goal of this data migration framework is to help companies migrate their massive data stored in relational databases to cloud platforms through web access. We propose a series of rules and constraints on the implementation of the framework, by which the users can conduct massive data migration with a multi-demand perspective.

Note: The cloud platforms mentioned in the document refer to the Hadoop platform by default. All standards on the operations and the environment of the framework refer to web state by default.

3.1. System Architecture Diagram

Figure 1 shows the working diagram of the framework.

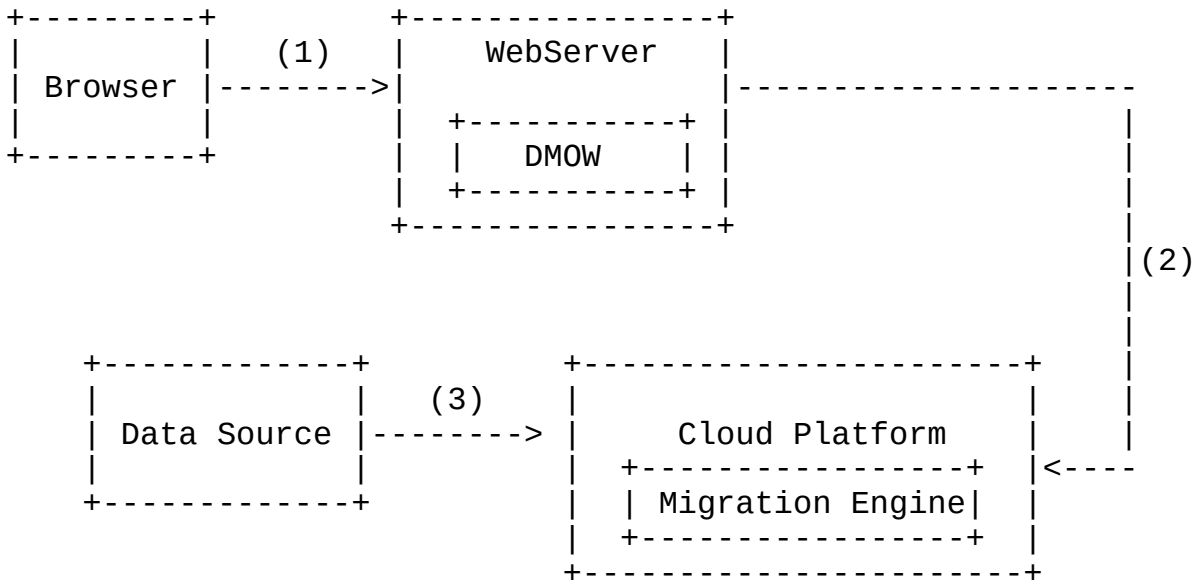

Figure 1:Reference Architecture

The workflow of the framework is as follows:

Step (1) in the figure means that users submit the requisition of data migration to DMOW through browser(the requisition includes data source information, target cloud platform information, and related migration parameter settings);

Step (2) in the figure means that DMOW submits user’s request information of data migration to cloud platform’s migration engine;

Step (3) in the figure means that the migration engine performs data migration tasks based on the migration requests it receives to migrate data from relational database to cloud platform;

3.2. Source and Target of Migration

3.2.1. The Data Sources of Migration

This framework MUST support data migration between relational databases and cloud platforms on web, and MUST meet the following requirements:

1. The framework supports to connect data sources in relational databases. The relational database MUST be at least one of the following:

   * SQLSERVER

   * MYSQL

   * ORACLE

2. This framework MUST support the dynamic perception of data information in relational databases under a normal connection, in other words :

   * It MUST support dynamic awareness of all tables in a relational database;

   * It MUST support dynamic awareness of all columns corresponding to all tables in a relational database;

3.2.2. The Connection Testing of Relational Data Sources

Before conducting data migration, the framework MUST support testing the connection to the data sources that will be migrated, and then decide whether to migrate.

3.2.3. The Target Storage Container of Data Migration

This framework MUST allow users to migrate large amounts of data from a relational database to the following at least two types of target storage containers:

- HDFS
- HBASE
- HIVE

3.2.4. Specifying Target Cloud Platform

This framework MUST allow an authorized user to specify the target cloud platform to which the data will be migrated.

3.2.5. Data Migration to third-party Web Applications

This framework SHALL support the migration of large amounts of data from relational databases to one or multiple data containers for third-party Web applications. The target storage containers of the third-party Web application systems can be:

- MONGODB
- MYSQL
- SQLSERVER
- ORACLE

3.3. Type of Migrated Database

This framework is needed to meet the following requirements:

- It MAY support migrating the entire relational database to the cloud platform;
- It MAY support homogeneous migration (for example, migration from ORACLE to ORACLE);
- It MAY support heterogeneous migrations between different databases (for example, migration from ORACLE to SQLServer);
- It SHALL support the migration to the MONGODB database;

- It’s OPTIONAL that If the migration process is interrupted, it is needed to support automatic restart of the migration process and continue the migration from where it left off; Additionally, the framework is needed to be able to support the user in the following manner to inform this abnormal interruption:
  - It MUST support popping up an alert box on the screen of the user;
  - It SHALL support notifying users by email;
  - It’s OPTIONAL to notify users by an Instant Messenger as We Chat or QQ;

3.4. Scale of Migrated Table

3.4.1. Full Table Migration

This framework MUST support the migration of all tables in a relational database to at least two types of target storage containers:

- HDFS
- HBASE
- HIVE

3.4.2. Single Table Migration

This framework MUST allow users to specify a single table in a relational database and migrate it to at least two types of target storage containers:

- HDFS
- HBASE
- HIVE

3.4.3. Multi-table migration

This framework MUST allow users to specify multiple tables in a relational database and migrate to at least two types of target storage containers:

- HDFS

- HBASE
- HIVE

3.5. Split-by

This framework is needed to meet the following requirements on split-by.

3.5.1. Single Column

1. The framework MUST allow the user to specify a single column of the data table (usually the table’s primary key), then slice the data in the table into multiple parallel tasks based on this column, and migrate the sliced data to one or more of the following target data containers respectively:
    * HDFS
    * HBASE
    * HIVE

    The specification of the data table column can be based on the following methods:

    + Users can specify freely;
    + Users can specify linearly;
    + Users can select an appropriate column for the segmentation based on the information entropy of the selected column data;

2. The framework SHALL allow the user to query the boundaries of the specified column in the split-by, then slice the data into multiple parallel tasks and migrating the data to one or more of the following target data containers:
    * HDFS
    * HBASE
    * HIVE

3.5.2. Multiple Column

This framework MAY allow the user to specify multiple columns in the data table to slice the data linearly into multiple parallel tasks and then migrate the data to one or more of the following target data containers:

- HDFS
- HBASE
- HIVE

3.5.3. Non-linear Segmentation

It’s OPTIONAL that this framework is needed to support non-linear intelligent segmentations of data for one or more columns and then migrate the data to one or more of the following target data containers:

The non-linear intelligent segmentations refer to:

* Adaptive segmentation based on the distribution(density)of the value of numerical columns;
* Adaptive segmentation based on the distribution of entropy of subsegments of a column;
* Adaptive Segmentation Based on Neural Network Predictor;

The target data container includes:

* HDFS
* HBASE
* HIVE

3.6. Conditional Query Migration

This framework SHALL allow users to specify the query conditions, then querying out the corresponding data records and migrating them.

3.7. Dynamic Detection of Data Redundancy

It’s OPTIONAL that the framework is needed to allow users to add data redundancy labels and label communication mechanisms, then it detects

redundant data dynamically during data migration to achieve non-redundant migration.

The detection of data redundancy can be based on the following methods:

- Redundancy detection based on data tables;
- Redundancy detection based on data files;

3.8. Data Migration with Compression

During the data migration process, the data is not compressed by default. This framework MUST support at least one of the following data compression encoding formats, allowing the user to compress and migrate the data:

- GZIP
- BZIP2

3.9. Updating Mode of Data Migration

3.9.1. Appending Migration

This framework SHALL support the migration of appending data to existing datasets in HDFS.

3.9.2. Overwriting the Import

When importing data into HIVE, the framework SHALL support overwriting the original dataset and saving it.

3.10. The Encryption and Decryption of Data Migration

This framework is needed to meet the following requirements:

- It MAY support data encryption at the source, and then the received data should be decrypted and stored on the target platform;
- It MUST support the authentication when getting data migration source data;
- It SHALL support the verification of identity and permission when accessing the target platform of data migration;

- During the process of data migration, it SHOULD support data consistency;
- During the process of data migration, it MUST support data integrity;

3.11. Incremental Migration

The framework SHOULD support incremental migration of table records in a relational database, and it MUST allow the user to specify a field value as "last_value" in the table in order to characterize the row record increment. Then, the framework SHOULD migrate those records in the table whose field value is greater than the specified "last_value", and then update the last_value.

3.12. Real-Time Synchronization Migration

The framework SHALL support real-time synchronous migration of updated data and incremental data from a relational database to one or many of the following target data containers:

- HDFS
- HBASE
- HIVE

3.13. The Direct Mode of Data Migration

This framework MUST support data migration in direct mode, which can increase the data migration rate.

Note:This mode supports only for MYSQL and POSTGRESQL.

3.14. The Storage Format of Data files

This framework MUST support to save the migrated data within at least one of following data file formats:

- SEQUENCE
- TEXTFILE
- AVRO

3.15. The Number of Map Tasks

This framework MUST allow the user to specify a number of map tasks to start a corresponding number of map tasks for migrating large amounts of data in parallel.

3.16. The selection on the elements in a table to be migrated column

- The specification of columns

  This framework MUST support the user to specify the data of one or multiple columns in a table to be migrated.

- The specification of rows

  This framework SHOULD support the user to specify the range of rows in a table to be migrated.

- The composition of the specification of columns and rows

  This framework MAY support optionally the user to specify the range of rows and columns in a table to be migrated.

3.17. Visualization of Migration

3.17.1. Dataset Visualization

After the framework has migrated the data in the relational database,,it MUST support the visualization of the dataset in the cloud platform.

3.17.2. Visualization of Data Migration Progress

The framework SHOULD support to show dynamically the progress to users in graphical mode when migrating.

3.18. Smart Analysis of Migration

The framework MAY provide automated migration proposals to facilitate the user’s estimation of migration workload and costs.

3.19. Task Scheduling

The framework SHALL support the user to set various migration parameters(such as map tasks,the storage format of data files,the type of data compression and so on) and task execution time, and then to perform the schedule off-line/online migration tasks.

3.20. The Alarm of Task Error

When the task fails, the framework MUST at least support to notify stakeholders through a predefined way.

3.21. Data Export From Cloud to RDBMS

3.21.1. Data Export Diagram

Figure 2 shows the framework's working diagram of exporting data.

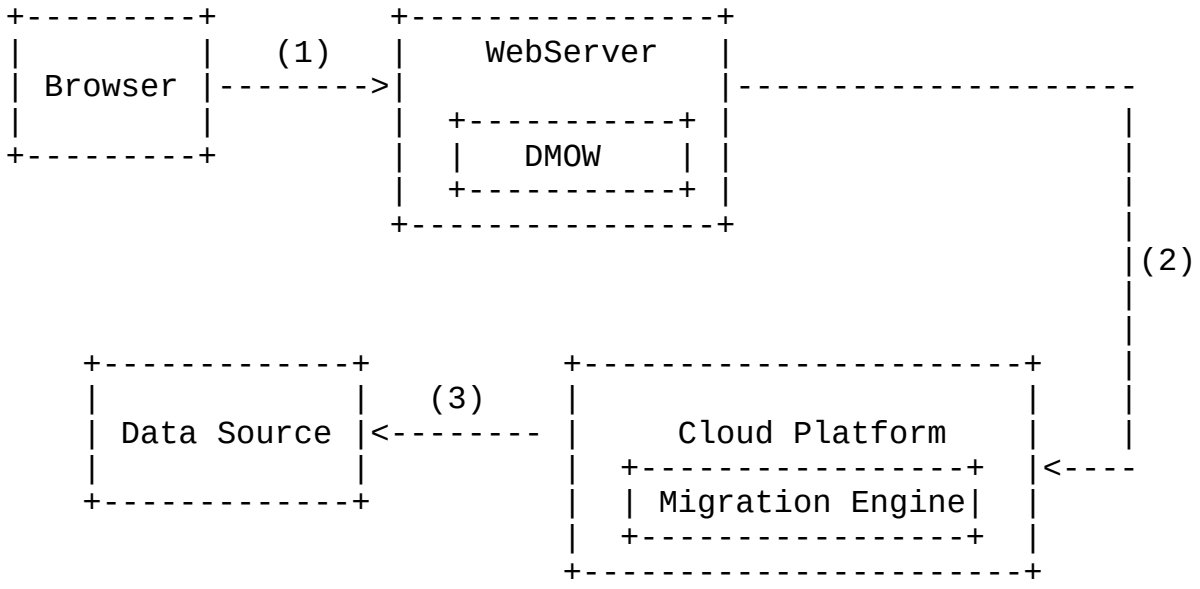

Figure 2:Reference Diagram

The workflow of exporting data through the framework is as follows:

Step (1) in the figure means that users submit the requisition of data migration to DMOW through browser(the requisition includes cloud platform information,the information of target relational database, and related migration parameter settings);

Step (2) in the figure means that DMOW submits user's request information of data migration to cloud platform's migration engine;

Step (3) in the figure means that the migration engine performs data migration tasks based on the migration requests it receives to migrate data from cloud platform to relational database;

3.21.2. Full Export

The framework MUST at least support exporting data from HDFS to one of following relational databases:

- SQLSERVER
- MYSQL
- ORACLE

The framework SHALL support exporting data from HBASE to one of following relational databases:

- SQLSERVER
- MYSQL
- ORACLE

The framework SHALL support exporting data from HIVE to one of following relational databases:

- SQLSERVER
- MYSQL
- ORACLE

3.21.3. Partial Export

The framework SHALL allow the user to specify data range of keys on the cloud platform and export the elements in the specified range to a relational database. Exporting into A Subset of Columns.

3.22. The Merger of Data

The framework SHALL support merging data in different directories in HDFS and store them in a specified directory.

3.23. Column Separator

The framework MUST allow the user to specify the separator between fields in the migration process.

3.24. Record Line Separator

The framework MUST allow the user to specify the separator between the record lines after the migration is complete.

3.25. The Mode of Payment

1. One-way payment mode

   * In the framework by default, users SHALL to pay for downloading data from the cloud platform.It is free when uploading data from a relational database to a cloud platform;

   * In the framework, users SHALL pay for uploading data from a relational database to a cloud platform.It is free when downloading data from the cloud;

2. Two-way payment mode

   In the framework, the users of the data migration process between the relational database and the cloud platform all SHALL pay a fee;

3.26. Web Shell for Migration

The framework provides following shells for character interface to operate through web access.

3.26.1. Linux Web Shell

The framework SHALL support Linux shell through web access, which allows users to perform basic Linux command instructions for the configuration management of the data migrated on web.

3.26.2. HBase Shell

The framework SHALL support hbase shell through web access, which allows users to perform basic operations such as adding, deleting, and deleting to the data migrated to hbase through the web shell.

3.26.3. Hive Shell

The framework SHALL support hive shell through web access, which allows users to perform basic operations such as adding, deleting, and deleting to the data migrated to hive through the web shell.

3.26.4. Hadoop Shell

The framework SHALL support the Hadoop shell through web access so that users can perform basic Hadoop command operations through the web shell.

3.26.5. Spark Shell

The framework SHALL support spark shell through web access and provide an interactive way to analyze and process the data in the cloud platform.

3.26.6. Spark Shell Programming Language

In spark web shell, the framework SHALL support at least one of the following programming languages:

- Scala
- Java
- Python

4. Security Considerations

The framework SHOUD support for the security of the data migration process. During the data migration process, it should support encrypt the data before transmission, and then decrypt it for storage in target after the transfer is complete. At the same time, it must support the authentication when getting data migration source data and it shall support the verification of identity and permission when accessing the target platform.

5. IANA Considerations

This memo includes no request to IANA.

6. References

6.1. Normative References

[RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, DOI 10.17487/RFC2026, October 1996, <https://www.rfc-editor.org/info/rfc2026>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, DOI 10.17487/RFC2578, April 1999, <https://www.rfc-editor.org/info/rfc2578>.

6.2. Informative References

[RFC2629] Rose, M., "Writing I-Ds and RFCs using XML", RFC 2629, DOI 10.17487/RFC2629, June 1999, <https://www.rfc-editor.org/info/rfc2629>.

[RFC4710] Siddiqui, A., Romascanu, D., and E. Golovinsky, "Real-time Application Quality-of-Service Monitoring (RAQMON) Framework", RFC 4710, DOI 10.17487/RFC4710, October 2006, <https://www.rfc-editor.org/info/rfc4710>.

[RFC5694] Camarillo, G., Ed. and IAB, "Peer-to-Peer (P2P) Architecture: Definition, Taxonomies, Examples, and Applicability", RFC 5694, DOI 10.17487/RFC5694, November 2009, <https://www.rfc-editor.org/info/rfc5694>.

6.3. URL References

[hadoop] The Apache Software Foundation, "http://hadoop.apache.org/".

[hbase] The Apache Software Foundation, "http://hbase.apache.org/".

[hive] The Apache Software Foundation, "http://hive.apache.org/".

[idguidelines] IETF Internet Drafts editor, "http://www.ietf.org/ietf/1id-guidelines.txt".

[idnits] IETF Internet Drafts editor, "http://www.ietf.org/ID-Checklist.html".

[ietf] IETF Tools Team, "http://tools.ietf.org".

[ops] the IETF OPS Area, "http://www.ops.ietf.org".

[spark] The Apache Software Foundation,
"http://spark.apache.org/".

[sqoop] The Apache Software Foundation,
"http://sqoop.apache.org/".

[xml2rfc] XML2RFC tools and documentation,
"http://xml.resource.org".

Authors' Addresses

Can Yang (editor)
South China University of Technology
382 Zhonghuan Road East
Guangzhou Higher Education Mega Centre
Guangzhou, Panyu District
P.R.China

Phone: +86 18602029601
Email: cscyang@scut.edu.cn

Yu Liu (editor)
South China University of Technology
382 Zhonghuan Road East
Guangzhou Higher Education Mega Centre
Guangzhou, Panyu District
P.R.China

Email: 201621032214@scut.edu.cn

Cong Chen
Inspur
163 Pingyun Road
Guangzhou, Tianhe District
P.R.China

Email: chen_cong@insour.com

Ge Chen
GSTA
No. 109 Zhongshan Road West, Guangdong Telecom Technology Building
Guangzhou, Tianhe District
P.R.China

Email: cheng@gsta.com

Yukai Wei
Huawei
Putian Huawei base
Shenzhen, Longgang District
P.R.China

Email: weiyukai@huawei.com