(ICBMS-3 specifics) The IoT Platform for Virtual Things, Distributed Autonomous Intelligence and Data Federation/Analysis

SDA Framework Open Source Manual

Summary: It describes how to develop by applying the SDA framework.

PINEONE

Document approval

If approved as an attached document of the main document, it is replaced with the signature of the main document.

Company : PINEONE Communications Co., Ltd.

_____

Yunho Bae Date

Development Team

Supervisor : PINEONE Communications Co., Ltd.

_____

Taecheol Kim Date

Development Team

Revised history

Revision number	Revision page and contents	The date of revision
0.1	1. Definition of Index	2016.11.28
1.0	Edit content	2016.12.28
2.0	Modify and add content (addition of composition and SDA module content)	2018.01.10

Index

Development Guide

1. Summary

Oasis (Open-source Architecture Semantic Iot Service-platform) project aims to develop an open source, intelligent IoT(Internet of Things) service platform based on international standards.

The Oasis project will be provided as open source as the outcome of the "(ICBMS-3 specifics) The IoT Platform for Virtual Things, Distributed Autonomous Intelligence and Data Federation/Analysis" as a newly supported by broadcasting and communication technology development project(2015).

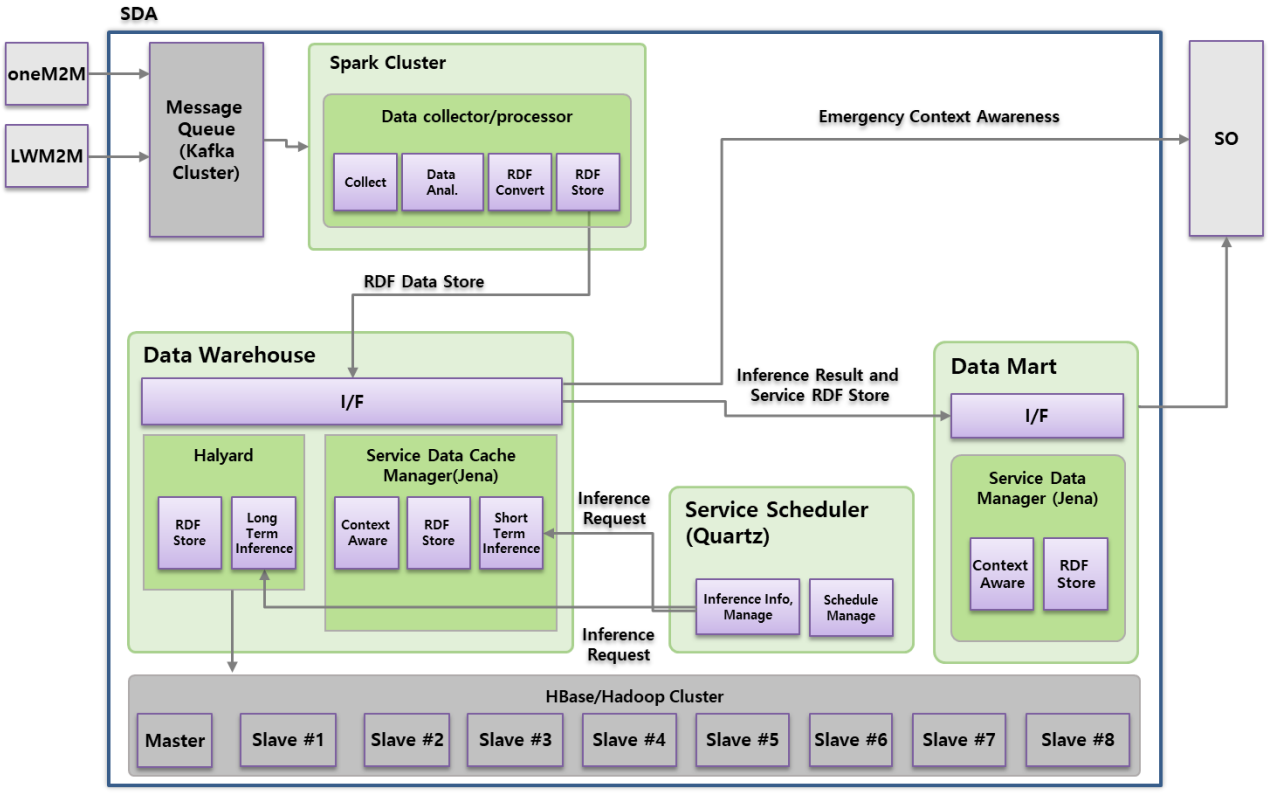
The Oasis project will continue to grow as an open source based on the open source community.

1.1 SDA (Semantic Data Analysis)

SDA is a data analysis framework that supports the ability to collect, analyze, and provide sensor/device data and legacy data occurred in IoT environments.

1.2 SDA (Semantic Data Analysis) Diagram

● SDA Framework Diagram



2. Main Functions

- Data collection based on the OneM2M standard platform
- Complies with Semantic Web standards such as RDF/OWL
- Semantic Annotation
- Apply reference model for IoT domain such as sensor network ontology (SSNO) and context awareness (DUL)
- Inference function for real-time context awareness

3. Development Method

3.1 Quick Start

For users who is approached Oasis SDA FrameWork at first, it guides you to download the source and test easily.

The SDA Server test can be proceeded in the following order.

1. MariaDB, JDK download and install
2. Oasis SDA source download
3. MariaDB and create table
4. SDA build
5. SDA setting and Web Module run

3.1.1 *Requirements*

- JDK 7+
- MariaDB 10.1.X+
- Maven 3.3.X+
- Tomcat 7.0.X+
- Windows / Linux

3.1.2 *Follow*

3.1.2.1 MariaDB download and install, JDK download and install

- MariaDB install guide
- JDK install guide

3.1.2.2 Oasis SDA source download

- Download the SDA source and installation files from the release page.

3.1.2.3 MariaDB basic setting

- Run the mariadb script file (mariadb_script.txt) downloaded from the download.

3.1.2.4 SDA build

- Build the SDA source in Eclipse (using Maven).downloaded from the release page.
- For information on how to build an SDA source, see the Source Build Method page.

3.1.2.5 SDA setting and run

- Modify the SDA settings by opening the sda-common/resources/system.properties file of the downloaded source.
- For how to set SDA, see the SDA Server Setting Method page.

3.2 SDA server setting

It describes how to set up the SDA server.

The settings are written in the properties file and reside in the resources folder of the sda-common folder.

system.properties

The meanings of the setting items are as follows.

- jena operations
  com.pineone.icbms.sda.knowledgebase.host=XXX.XXX.XXX.XXX
  com.pineone.icbms.sda.knowledgebase.public.host=XXX.XXX.XXX.XXX

- fuseki operations
  com.pineone.icbms.sda.knowledgebase.sparql.endpoint=http://XXX.XXX.XXX.XXX:PORT/icbms c
  om.pineone.icbms.sda.knowledgebase.host.port=PORT
  com.pineone.icbms.sda.knowledgebase.uri=http://www.iotoasis.org

- SI operations
  com.pineone.icbms.sda.mongodb.server=XXX.XXX.XXX.XXX
  com.pineone.icbms.sda.mongodb.port=PORT com.pineone.icbms.sda.mongodb.db=DB NAME

- Process number of data collection per one time
  com.pineone.icbms.sda.mongodb.read_limit=20000

- DB information operations
  com.pineone.icbms.sda.ss.db.server=XXX.XXX.XXX.XXX com.pineone.icbms.sda.ss.db.port=PORT
  com.pineone.icbms.sda.ss.db.name.timetable=DB NAME
  com.pineone.icbms.sda.ss.db.name.device= DB NAME com.pineone.icbms.sda.ss.db.user=USER
  ACCOUNT com.pineone.icbms.sda.ss.db.pass=PASSWORD

- s-post location of fuseki
  com.pineone.icbms.sda.triple.regist.bin=/fuseki DIRECTORY/bin/s-post

- triple file repository location
  com.pineone.icbms.sda.triple.save_path=/triple REPOSITORY FOLDER /triples

- Callback URI of the SDA operation server
  com.pineone.icbms.sda.si.notification_uri=http://XXX.XXX.XXX.XXX:PORT/sda/subscribe/callback

- SI operation (for description setting)
  com.pineone.icbms.sda.si.subscription_uri=http://XXX.XXX.XXX.XXX:PORT NUMBER /PATH/

- SO operation (for callback)
  com.pineone.icbms.sda.so.callback_result_uri=http://XXX.XXX.XXX.XXX: PORT NUMBER /PATH/

- riot mode
  --skip : riot not applied
  --check : Check the whole file and extract the errored part —> Use during development
  --validate : When scanning a file, it stops parsing immediately if there is an error in the grammar and turn off in case of error
  com.pineone.icbms.sda.riot.bin=/jena INSTALL DIRECTORY/bin/riot
  com.pineone.icbms.sda.riot.mode=--skip com.pineone.icbms.sda.riot.result.save_path=/riot RESULT REPOSITORY FOLDER/riot-result

- Number of kafka threads
  com.pineone.icbms.sda.kafka.thread.count=3

- Specify the minus time in milliseconds to be used when determining the reference time to obtain lastestedContentInstance.
  If it is valid from (Current time - 10 seconds), specify a value of 1000*10.
  com.pineone.icbms.sda.init.adjust.ms=10000

- Db access information to perform statistical query
com.pineone.icbms.sda.stat.db.server=XXX.XXX.XXX.XXX com.pineone.icbms.sda.stat.db.port=PORT NUMBER com.pineone.icbms.sda.stat.db.name=DB NAME com.pineone.icbms.sda.stat.db.user= USER ACCOUNT com.pineone.icbms.sda.stat.db.pass=PASSWORD

-mongodb Information (for statistics)
com.pineone.icbms.sda.mongo.db.server=XXX.XXX.XXX.XXX
com.pineone.icbms.sda.mongo.db.port=PORT NUMBER com.pineone.icbms.sda.mongo.db.name=DB NAME com.pineone.icbms.sda.mongo.db.collection.name=collection NAME

3.3 SDA Framework Server Build

It describes how to build the downloaded SDA Framework server source using Eclipse.

The requirements for Build are as follows.

3.3.1 *Requirements*

- JDK 7+
- Windows / Linux
- Eclipse
- Maven

Build run sequence is as follows.

1. Import source into Eclipse
2. Compile the source using Maven
3. Add and modify configuration files
4. Run the sda-web module

3.3.1.1 Source import to Eclipse

- Open the Import window by selecting File/Import menu from the menu and select Maven Project. And select the folder where the source is saved and import.

3.3.1.2 Adding and modifying configuration files

- Modify the configuration file (system.properties) downloaded from the Release page.
- Modify the configuration file according to local environment such as DB setting. For more information on the configuration file, see the SDA Server Settings page.

3.3.1.3 Build with Maven

- Run Maven Build with Alt + Shift + X, M.
- Enter package and run the build

3.3.1.4 Running the SDA Web Module

- Rename and copy the target folder(sda-web/target/sda-web-2.0.war) to tomcat's webapps as sda.war and start tomcat.

4. Module

The SDA Framework consists of three server modules.

4.1 sda-web : Web module for providing RESTFul API and web service etc.

1. sda-web/com.pineone.icbms.sda.itf : CRUD class for cm, ci
   A. sda-web/com.pineone.icbms.sda.itf.ci : CRUD for ci
   B. sda-web/com.pineone.icbms.sda.itf.ci : CRUD for cm
   C. sda-web/com.pineone.icbms.sda.itf.cmi : CRUD for cm/ci(mainly if you need to use it together)

2. sda-web/com.pineone.icbms.sda.logger : Specifying commonly applied Logs

3. sda-web/com.pineone.icbms.sda.sch : Collect data from SI or perform internal context scheduling
   A. sda-web/com.pineone.icbms.sda.sch.comm : Scheduler common class
   B. sda-web/com.pineone.icbms.sda.sch.controller : Scheduler controller class
   C. sda-web/com.pineone.icbms.sda.sch.dao : The scheduler dao class
   D. sda-web/com.pineone.icbms.sda.sch.service : Scheduler service class
   E. sda-web/com.pineone.icbms.sda.sch.service.CollectDataFromSIJobService : Data collection from SI
   F. sda-web/com.pineone.icbms.sda.sch.service. Other classes: Classes for internal contexts

4. sda-web/com.pineone.icbms.sda.sf : Class for providing context aware RESTFul API
5. sda-web/com.pineone.icbms.sda.subscribe : Class for providing SI subscribe API (currently not used)

6. sda-web/resources/config : Configuration file for spring framework (*.xml)
7. sda-web/resources/mapper : Query file for mybatis (*.xml)

4.2 sda-client : Modules that have a client program and runs independently

1. sda-client/com.pineone.icbms.sda : Import the onem2m data registered in the kafka broker into a triple data file and store it in the triple repository using fuseki's s-put.sh
2. sda-client/com.pineone.icbms.sda.hbase.onem2m : onem2m package that collects data, but is not used

3. sda-client/com.pineone.icbms.sda.kafka.onem2m.AvroOneM2MDataSubscribe : Take onem2m data from kafka broker and make it into triple data file and save it in triple repository

4.3 **sda-common :** sda's core module, which includes common functions and core functions

1. sda-common/com.pineone.icbms.sda.comm : dto, exception definition class, commonly used in sda-web or sda-client, kafka avro transmission class
   A. sda-common/com.pineone.icbms.sda.comm.kafka.onem2m : Class used by AvroOneM2MDataSubscribe.java file of sda-client
   B. sda-common/com.pineone.icbms.sda.comm.kb : module to convert onem2m data to triple data
   C. sda-common/com.pineone.icbms.sda.comm.sf : Modules that include dao, dto, and service for context awareness
2. sda-common/resources/mariadb : The db connection class and information used to look up log and context model information in sda-common modules
3. sda-common/resources/system.properties : A config file that contains commonly used configuration values

5. Q&A