JakSAFE JakSERVICE Post Processing Technical Documentation

Revision History: Version 1.0
Last Updated: 24 April 2015

inteligensi risiko
your partner in catastrophe risk analysis

PT Inteligensi Risiko
Office8, Level 18-A
Sudirman Central Business District
Jl. Jendral Sudirman Kav. 52-53
Jakarta Selatan, 12190
Indonesia
+62 21 2955 7217
+62 21 2955 7218

Disiapkan untuk:

WORLD BANK GROUP
GFDRR
Global Facility for Disaster Reduction and Recovery

Contents

Document Information

Status	Initial Release	Document Date	24 April 2015
Author	Abdul Somat Budiaji		

Authorisation

Reviewed By		Date	
Approved By		Date	
Reviewed By		Date	
Approved By		Date	
Reviewed By		Date	
Approved By		Date	
Reviewed By		Date	

Change History

Version	Date	Authors	Summary of Changes
1.0	24 April 2015	Abdul Somat Budiaji	Initial release

1 Introduction

1.1 Purpose

The purpose of this document is to give technical information about the JakSAFE JakSERVICE post processing design and implementation.

1.2 Intended Audience

The intended audience of this technical document are the following:

- JakSAFE and JakSERVICE development team
- Future contributors to the project

1.3 Scope

This document will describe the design and implementation of the JakSAFE JakSERVICE post processing. To fully comprehend the implementation some basic knowledge is required in the following subjects:

- Python programming
- Python binding for QGIS API
- Python mysql binding
- Pandas python library
- Numpy python library
- Matplotlib python library

1.4 Overview

The JakSAFE JakSERVICE post processing is a python application that process data aggregated from flood sources to calculate damage and loss assessment (DALA). There are two types of post processing that is automatic post processing and ad hoc post processing. Automatic post processing is executed every six hours while ad hoc post processing is executed at the user's choice.

1.5 Features

The following are the features of the JakSAFE JakSERVICE post processing:

- Calculate Damage and Loss Assesment
- Generate Damage and Loss Assesment Report

1.6 Software Components

As of version 1.0 the JakSAFE JakSERVICE post processing comprise of the following application/software/library components to provide its functionalities:

No.	Name	Function	Version	Website
1.	Python	Main development language	2.7.6	https://www.python.org/
2.	MySQL connector python	Python binding for MySQL database	1.1.6	http://dev.mysql.com/doc/connector-python/en/index.html

3.	Pandas	Data analysis	0.15.2	http://pandas.pydata.org/
4.	Numpy	Numerical function and calculation support	1.8.2	http://www.numpy.org/
5.	QGIS API	Spatial operation and interface to QGIS operation	2.6	http://qgis.org/api/
6.	Maplotlib	Plotting data and report generation	1.3.1	http://matplotlib.org/

Figure 1 JakSAFE webapp software components

2 Directory Structure

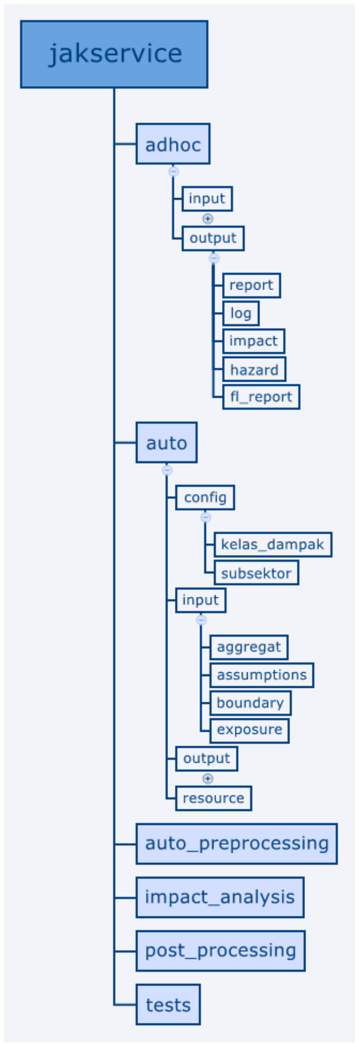

Figure 2 Directory Structure of Jakservice

Figure 2 describes the directory structure of JakSERVICE program. At the root there is **Jakservice** directory. Below **Jakservice** there are **adhoc** and **auto** directory which indicate two kinds of DALA calculation, **auto_preprocessing** which stores source code for auto preprocessing function,

impact_analysis which stores source code for impact analysis function, **post_processing** which store source code for post processing function, and **tests** which store code for test purpose.

Both **adhoc** and **auto** have **input** and **output** directories. **Input** directory contains all necessary input file for JakSERVICE program to run properly. **Output** directory contains all the result of JakSERVICE program. In addition, **auto** directory contains **config** directory and **resource** directory. These directories are used also by the adhoc process.

2.1 Files in JakSERVICE

Directory/filename	Description
./run_dalla_auto.py	Main script to run JakSERVICE auto DALA calculation
./run_dalla_adhoc.py	Main script to run JakSERVICE adhoc DALA calculation
./global_conf.cfg	Global configuration file of JakSERVICE program, described more detail in section 2.1.1

Figure 3 Files in directory jakservice

2.1.1 Global Configuration File

Global configuration file in JakSERVICE program is stored in global_conf.cfg file. Figure 4 summarize options in global_conf.cfg file.

Option	Description
SECTION : database_configuration	
url_address	
user	
passwd	
database_name	
port	
table_name_event	
table_raw_name_event	
table_name_autocalc	
table_name_adhoc_calc	
SECTION : file_input	
input_kelas	
input_boundary_layer	
input_building_exposure	
input_road_exposure	
SECTION : file_output	
output_rw_report	
output_rt_report	
output_hazard	
output_building_exposure	
output_road_exposure	
SECTION : qgis_conf	
qgis_install_path	
SECTION : dims_conf	
url_dims	
SECTION : folder_conf	
project_folder	

auto_folder	
adhoc_folder	
SECTION : directory	
resource	Location of resource directory
assumption	Location of assumption directory
aggregate	Location of aggregate directory
log	Location of log auto output directory
impact	Location of impact auto directory
report	Location of report auto directory
hazard	Location of hazard auto directory
log_adhoc	Location of log adhoc directory
impact_adhoc	Location of impact adhoc directory
report_adhoc	Location of report adhoc directory
hazard_adhoc	Location of hazard adhoc directory
subsektor	Location of parent directory of subsektor.py
SECTION : subsektor	
subsektor	Comma separated list containing subsector that will include in DALA calculation. Subsektor name should be capital

Figure 4 List of options in global_conf.cfg file

2.2 Files in auto and adhoc

Basically, directory structure inside **auto** and **adhoc** is the same, except that adhoc does not have **config** and **resource** directories. Instead cover both directories, we will only cover **auto** directory.

2.2.1 Config Directory

- Kelas_dampak directory
  Kelas_dampak directory contains configuration about flood class
- Subsektor directory
  File subsektor.py is located within this directory. This file acts as a configuration for asset used in DALA calculation. This file consists of python dictionary whose key are assets avalaible to DALA calculation. Each key corresponds to a python dictionary with structure as follows:

Key	Description
SEKTOR	Sector of the asset
SUBSEKTOR	Subsector of the asset
ASET	The asset
IMPACT	Impact file to be used in DALA calculation
DALA	The type of DALA calculation used. There are various DALA calculation type that has been explain in great detail in the Functional Specification Document.

Figure 5 Description of Each Asset Configuration

2.2.2 Input Directory

- Aggregat directory
  Aggregate directory contain aggregate data.

- Assumptions directory
  Assumptions directory contain assumption that is used in DALA calculation.
- Boundary directory
  Boundary directory contain administrative shapefile for DKI Jakarta provinces down to RW adminstrative level.
- Exposure Directory
  Exposure Directory contains asset shapefiles spread over DKI Jakarta provinces covering buildings and roads.

2.2.3 Output Directory

- Fl_report directory
  Fl_report directory contains summarize of flood report coming from DIMS server
- Hazard directory
  Hazard directory contains directories whose name follows format as **YYYYMMDDHHMMSS_ YYYYMMDDHHMMSS**. This kind of format is seen many times in JakSERVICE result. So, we will not explain again next time. The first timestamp is always time-0 (request time) and the second timestamp is always time-1 (process runtime). Inside those directory are hazard shapefile named **YYYYMMDDHHMMSS_ YYYYMMDDHHMMSS_hazard**. Hazard shapefile is essentially a boundary shapefile that is embedded with flood class information.
- Impact directory
  Impact directory directory contains directories whose name follows format as **YYYYMMDDHHMMSS_ YYYYMMDDHHMMSS**. Inside those directory lie two directories named **shapefile** and **summary**. **shapefile** directory consists of impact shapefile. Impact shapefile is a exposure shapefile that is embedded with flood class information. Impact shapefile's name is **YYYYMMDDHHMMSS_ YYYYMMDDHHMMSS_building** and **YYYYMMDDHHMMSS_ YYYYMMDDHHMMSS_road**. **summary** directory consists of impact shapefile summary saved in csv format that is divided into three types: agg_impact.csv, osm_impact.csv, and osm_road_impact.csv.
- Log directory
  Log files of Jakservice process is generated inside log directory. Log file name formal follows **dala_YYYYMMDDHHMMSS_ YYYYMMDDHHMMSS.log**.
- Report directory
  Report directory contains result of calculation in csv file and final report in pdf format.

2.2.4 Resource Directory

This directory contains files that is essential to the process of generating DALA report, that is logo file of JakSAFE named jaksafe1.png.

2.3 Post Processing Source Code

Directory/File	Description
./adhoc.py	Script that is imported from run_dalla_adhoc.py, main interface to post processing adhoc
./aggregate.py	Interface for aggregate files
./asumsi.py	Interface for assumption files
./config.py	Interface for various configuration

./dala.py	Implementation of various type of DALA calculation
./db.py	Interface to MySQL database
./error.py	Various custom exception class
./hazard.py	Interface to hazard shapefile
./post.py	Post Processing analysis
./report.py	Script to generate pdf report from DALA calculation result
./run.py	Script that is imported from run_dalla_auto.py, main interface to post processing auto
./shape.py	Interface to QGIS functionality, convert shapefile to pandas data frame
./summary.py	Script to summarize DALA calculation result, some kind of preprocessing helper for report.py
./tools.py	Various useful tool, for now it only contains zipper for hazard shapefile, impact shapefile, and DALA calculation result

Figure 6 List of python scripts to deliver JakSERVICE post processing function

3 Software Architecture

3.1 Post Processing

3.1.1 run.py AND adhoc.py

run.py and **adhoc.py** implement auto post processing process and adhoc post processing process respectively. Figure 7 depicts the two kinds of JakSERVICE post processing process.

Input (all the same for both run.py and adhoc.py):

- time_0: initial time
- time_1: end time
- list_subsektor : List of subsector in global configuration file
- last_row_id

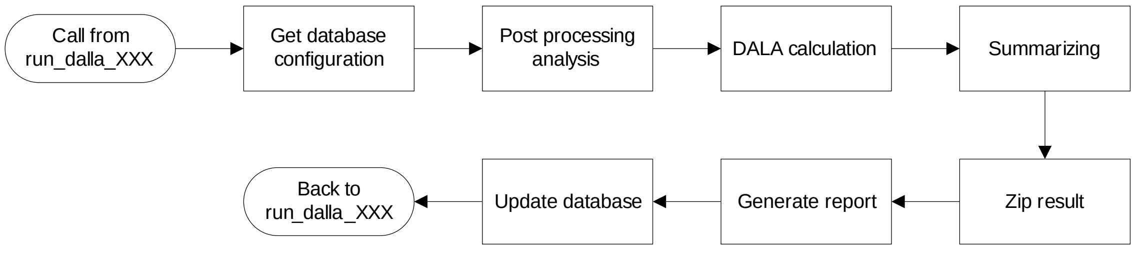

Figure 7 Jakservice Post Processing Process

3.1.2 config.py

config.py implement interfacing to various configuration files. There are three classes and one function.

1. class Subsektor
   Parameter : None
   This class implements interfacing with configuration in subsektor.py. This class have methods:
   - get_subsektor : get subsector provided asset as input
   - get_sektor : get sector provided asset and/or subsector as input
   - get_hazard : get what hazard shapefile to be used provided asset as input
   - get_impact : get impact summary file to be used provided asset as input
   - get_dala : get what kind of DALA calculation to be used provided asset as input
   - get_list_asset : get list of assets provided subsector as input
2. class Path
   Parameter : time_0, time_1, tipe [default is auto]
   This class implement interface to address of various files used in JakSERVICE application. This class has no method, but it has many attributes which represent various directory.
   - log_dir
   - resource_dir
   - impact_dir
   - output_dir
   - shp_impact_dir
   - shp_hazard_dir
   - summary_dir
3. class Database
   Parameter : None
   This class implement interfacing to database connection configuration with MySQL. It has no method and has only one attribute namely params_con which represents connection parameter to MySQL database.
4. function time_formatter
   This function changes time format from one to another.
   input:
   - time_input : input time to be changed
   - format_input : old time format
   - format_output : new time format
5. class ListSubsektor
   Parameter : None
   This class implement interfacing with list of subsectors in global configuration file.

3.1.3 db.py

db.py implement connection interface to MySQL database. It has only one class namely Dbase. The methods are:

1. close
   Close database connection
2. write

Insert data to table.
Input :

- table : table name which used to save the data
- data : data to be saved

3. update
Update table with data
Input :
    - table : table name to be updated
    - data : data used to update the table

3.1.4 post.py

post.py implement post processing analysis. It has only one class namely PostProc.

Input :

- time_0 : initial time
- time_1 : final time
- tipe : calculation type whether "auto" or "adhoc" [default to adhoc]

Methods :

1. building
Analyze post processing for impact building shapefile. Building method turn information inside impact building shapefile into pandas data frame.
2. road
Analyze post processing for impact road shapefile. Road method turn information inside impact road shapefile into pandas data frame. Both road and building method use helper module to turn shapefile into pandas dataframe namely **shape.py**.
3. aggregate
Analyze post processing for aggregate file. Aggregate file is a user defined input. It contains information about how much a certain asset located in certain area. Aggregate method returns pandas data frame that contains information about number of asset affected by flood in certain location. In the process aggregat method is helped by **aggregate.py** and **hazard.shp**.
4. analyze
Analyze method is the main method. It calls all three methods and saves the resulted data frame into three separated files which represent building, road, and aggregate.

3.1.5 shape.py

shape.py implement interfacing to QGIS functionality and shapefile. It has only one class namely Shape.

Input :

- shapefile : shape file to be processed

Methods :

- get_features
  Get features from layer in the shapefile. This method has one input which is request whose default value is none. Request is some kind of query in QGIS to determine which feature get selected.
- get_dataframe
  Get dataframe from the feature selected. It has one input which is list of argument that determine which attribute to be included in resulting pandas data frame.

3.1.6 aggregate.py

aggregate.py implement interfacing to aggregate input file. It has only one class namely Aggregate.py.

Input :

- input_file : aggregate input file

Methods :

- jumlah
  This method returns the total of an asset in a particular location
- aset_in
  This method get part of aggregate data based on criteria such as the type of asset, geolevel location.
- satu
  This method get the total of an asset and its location
- fix_format_rt
  This method fix RT name in order to conform to common naming standard

3.1.7 hazard.py

hazard.py implement interfacing to hazard shapefile. It has only one class namely Hazard.

Input :

- hazard_shp : hazard shapefile

Methods :

- percent
  This method return pandas data frame containing percentage area that is affected by flood.
- detail
  This method calculate percentage area affected by flood
- percent_agg
  This method return pandas data frame containing percentage area that is affected by flood.

3.1.8 dala.py

dala.py implements the DALA calculation used in JakSERVICE program. It has only one class namely Dala. In class Dala there is module **asumsi** which helps in providing assumption information about asset loss and damage for particular flood class.

Methods :

- calculate
  This is main method and it returns nothing. It accept input of list of subsector and choose apropriate dala calculation method then save the result to a csv file.
- dala_nol
  This dala calculation method is chosen if no asset is affected by flood.
- dala_satu
  This dala calculation is the deafult method when per unit asset information is availaible. The calculation simply multiplication of class at impact file times its associated assumption.
- dala_dua
  This dala calculation is used when the avalaible asset information is area. For now, this method is only for "tambak" and "kebersihan" assets.
- dala_tiga
  This dala calculation is used when the avalaible information is length. It is used for road ("jalan") asset.
- dala_empat
  This dala calculation is used for vehicle ("kendaraan") asset only.
- dala_lima
  This dala calculation is used for asset that has no unit nor length nor area information. It does not either have assumption information in asumsi_kerusakan.csv/asumsi_kerugian.csv file. Instead it has information aggregate loss/damage for a whole area.
- dala_enam
  This type of dala calculation is only for insurance ("asuransi") asset to calculate damage only.
- dala_asuransi
  This type of dala calculation is only for insurance ("asuransi") asset to calculate loss only.
- dala_per
  This method save dala calculation result to a file, grouped by subsector.

3.1.9 asumsi.py

asumsi.py implement interfacing to assumptions file. It has only one class namely Asumsi.

Methods :

- agregat
  This method is to get information in "agregat" assumption file
- asuransi
  This method is to get information in "asuransi" assumption file
- penetrasi_asuransi
  This method is to get information in "penetrasi asuransi" assumption file

3.1.10 summary.py

summary.py implement summarizing function of DALA calculation result in order to prevent difficulty in generating report.

Input :

- time_0 : initial time
- time_1 : final_time
- tipe : dala calculation type

Methods :

- normalize
- summarize
- total
- get_data_bar_page_1
- get_data_table_page_1
- get_data_bar_page_2
- get_data_pie_page_3
- get_data_table_page_4_and_5
- get_data_bar_page_6
- get_data_bar_per
- get_data_asuransi

3.1.11 tools.py

tools.py implement support tools for JakSERVICE. For now, it has only one class namely Zipper. Zipper class function is to zip output file from Jaksafe application.

3.1.12 report.py

report.py implement report generation for DALA calculation result.

3.1.13 error.py

error.py implement custom exception in Jaksafe application.

4 Running the JakSAFE JakSERVICE

Providing that input files and configuration files are set, Jakservice can be run using the following ways:

- Jakservice auto is run by execute
  python run_dalla_auto.py
- Jakservice adhoc is run by execute
  python run_dala_adhoc -s <start_time> -e <end_time>
  start_time and end_time format: YYYYMMDDHHMMSS (e.g. 20150424055959)