# lib库说明：

1. 文件名： ABMF\_proc.py

用途：用于处理类似vrqo\_QOConfig这样的文件，类似xml用<tag>做标记，但是又不是标准的xml。功能包括文件解析，文件回写，两个文件的差异合并。

函数：

1. 函数 parse\_QO\_config(ABMF\_file, split\_point)

用途：解析类似vrqo\_QOConfig的文件

输入：

ABMF\_file， str，文件名

split\_point， list(dict()…), 内容拆分的标记

输出：

ABMF\_list， list， 除标记包含之外的内容

ABMF\_partial\_fragment， 二级字典，每对标记包含的内容

1. 函数 write\_QO\_config：

用途：回写类似vrqo\_QOConfig的文件

输入：

ABMF\_list， list， 除标记包含之外的内容

ABMF\_partial\_fragment， 二级字典，每对标记包含的内容

ABMF\_file，str，回写的文件名（包含绝对路径）

输出：

成功/失败， bool

1. 函数 merge\_QO\_config\_fragment：

用途：按输入的条件合并类似vrqo\_QOConfig的文件

输入：

ABMF\_partial\_fragment\_s，二级字典， 除标记包含之外的内容（源文件）

ABMF\_partial\_fragment\_d， 二级字典，每对标记包含的内容（目的文件）

merge\_key\_list，dict(list[]…)，关键字信息，用户从源文件抽取相关片段，追加到目的文件中去

输出：

无

1. 函数 merge\_QO\_config：

用途：处理流程入口函数

输入：

ABMF\_partial\_fragment\_s，二级字典， 除标记包含之外的内容（源文件）

ABMF\_partial\_fragment\_d， 二级字典，每对标记包含的内容（目的文件）

split\_point， list(dict()…), 内容拆分的标记

merge\_key\_list，dict(list[]…)，关键字信息，用户从源文件抽取相关片段，追加到目的文件中去

输出：

无

1. 文件名： ini\_proc.py

用途：解析，修改，回写ini文件

函数：

1. 函数 parse\_ini

用途：解析ini文件

输入：

ini\_file， str，文件的路径

输出：

ini\_dict， 二级字典，{“section”:{“param”:”value”}}

1. 函数 write\_ini

用途：回写ini文件

输入：

data，二级字典，{“section”:{“param”:”value”}}

ini\_file， str，文件的路径

输出：

是否写文件成功， bool

1. 函数 modify\_ini

用途：根据输入条件修改ini文件，支持 add，replace，delete操作。

输入：

modify\_list，数据结构 [dict()…]，修改条件

ini\_file， str，文件的路径（读取和回写）

输出：

result\_list,，list(bool)，每个条件条件是否成功

1. 文件名： text\_proc.py

用途：解析，修改，回写text文件

函数：

1. 函数 parse\_text

用途：解析text文件

输入：

text\_file， str，文件的路径

encoding， str，编码方式

输出：

text\_list， 列表，文件中的每行是list中的一个元素

1. 函数 write\_text

用途：回写text文件

输入：

text\_list，list，list的每个元素是文件中的一行

text\_file， str，文件的路径

encoding， str，编码方式

输出：

是否写文件成功， bool

1. 函数 modify\_text

用途：根据输入条件修改text文件，并回写到源文件，支持 replace，append，insert，delete操作。

输入：

modify\_list，数据结构 [dict()…]，修改条件

text\_file， str，文件的路径（读取和回写）

encoding， str，编码方式

输出：

result\_list,，list(bool)，每个条件条件是否成功

1. 文件名： xml\_proc.py

用途：解析，修改，回写xml文件

函数：

1. 函数 parse\_xml\_rude

用途：解析xml文件

输入：

xml\_file， str，文件的路径

输出：

xml\_list， 列表，文件中的每行是list中的一个元素

1. 函数 write\_xml\_rude

用途：回写xml文件

输入：

xml\_list，list，list的每个元素是文件中的一行

xml\_file， str，文件的路径

输出：

是否写文件成功， bool

1. 函数 modify\_xml\_rude

用途：根据输入条件修改xml文件，并回写到源文件，支持 replace，add操作。

输入：

modify\_list，数据结构 [dict()…]，修改条件

xml\_file， str，文件的路径（读取和回写）

输出：

result\_list,，list(bool)，每个条件条件是否成功

1. 文件名： json\_proc.py

用途：解析，修改，回写ini文件

函数：

1. 函数 parse\_json

用途：解析json文件

输入：

json\_file， str，文件的路径

输出：

多级字典

1. 函数 write\_json

用途：回写json文件

输入：

data，多级字典

ini\_file， str，文件的路径

输出：

是否写文件成功， bool

1. 文件名： common\_function.py

用途：通用的处理函数集合

函数：

1. 函数 trim\_head\_tail(content='')

用途：剪切字符串前后的换行，回车换行，空格

输入：

content， str，字符串

输出：

处理后的字符串，str

1. 函数 extract\_data

用途：从字符串中抽取所需字段

输入：

content， str，原始字符串

rule， str，正则表达式

incise\_begin， int，对正则提取后的字符串进行截断，起始点坐标。

incise\_stop，int，对正则提取后的字符串进行截断，终止点坐标。

输出：

ABMF\_list， list， 除标记包含之外的内容

ABMF\_partial\_fragment， 二级字典，每对标记包含的内容

1. 文件名： compile\_proc.py

用途：编译C++的代码

函数：

1. 函数 compile\_cplus

用途：编译C++的代码

输入：

build\_path， str，执行编译的目录

输出：

是否执行成功，bool

1. 文件名： logger.py

用途：记录日志

函数：

1. 函数 write\_log

用途：记录日志

输入：

log\_content， str，日志内容

log\_file，str，日志文件路径

log\_oper\_mode，str，写文件的模式（新写/追加）

输出：

无

1. 函数 create\_logger

用途：创建python logging模块的句柄，方便用户直接调用写日志的方法

输入：

log\_name， str，日志的名字

log\_path，str，日志文件路径

log\_level，str，日志的级别

log\_format，str，日志的格式

输出：

Logger 句柄

1. 文件名： ssh\_inter\_act.py

用途：用ssh链接和其他服务器交互

函数：

1. 函数 SSHClient.\_\_init\_\_

用途：创建ssh对象

输入：

ip， str，对端ip地址

port，str，对端的port

user，str，对端的用户名

password，str，对端的密码

输出：

无

1. 函数 SSHClient.close

用途：关闭ssh链接

输入：

无

输出：

无

1. 函数 SSHClient.exe\_ssh\_cmd\_get\_echo

用途：执行ssh命令

输入：

os\_cmd， str，需要执行的命令

输出：

cmd\_echo，str，回显的内容

cmd\_echo\_err，str，执行命令的错误信息，如果执行成功则该值为’’

1. 文件名： sftp\_inter\_act.py

用途：用sftp链接和其他服务器交互，批量上传/下载文件

函数：

1. 函数 SFTPClient.\_\_init\_\_（继承自SSHClient类）

用途：创建sftp对象

输入：

ip， str，对端ip地址

port，str，对端的port

user，str，对端的用户名

password，str，对端的密码

输出：

无

1. 函数 SFTPClient.close

用途：关闭sftp链接

输入：

无

输出：

无

1. 函数 SFTPClient.sftp\_mkdir

用途：创建对端的目录

输入：

dir， str，需要创建的目录路径

输出：

成功/失败，bool

1. 函数 SFTPClient.sftp\_rmdir

用途：强制删除对端的目录

输入：

dir， str，需要删除的目录路径

输出：

成功/失败，bool

1. 函数 SFTPClient.sftp\_chdir

用途：进入对端的目录

输入：

dir， str，需要进入的目录路径

输出：

成功/失败，bool

1. 函数 SFTPClient.sftp\_get

用途：批量下载文件（包含多级子文件夹和文件）

输入：

remote\_path， str，对端的文件夹路径，或者具体的文件绝对路径

local\_path， str，本地的文件夹路径，或者具体的文件绝对路径（支持文件重命名）

输出：

无

1. 函数 SFTPClient.sftp\_put

用途：批量上传文件（包含多级子文件夹和文件）

输入：

local\_path， str，本地的文件夹路径，或者具体的文件绝对路径

remote\_path， str，对端的文件夹路径，或者具体的文件绝对路径（支持文件重命名）

输出：

无

1. 文件名： svn\_proc.py

用途：svn相关操作

函数：

1. 函数 SVNClient.\_\_init\_\_

用途：创建svn client对象

输入：

username，str，svn的用户名

password，str，svn的密码

输出：

无

1. 函数 SVNClient.svn\_checkout

用途：checkout代码

输入：

svn\_url，str，svn的url

local\_path，str，本地的路径

输出：

无

1. 文件名： timer\_mgmt.py

用途：时间处理的相关操作

函数：

1. 函数 CountdownTimer.\_\_init\_\_

用途：创建一个定时器对象（独立的线程）

输入：

timer\_start，int，定时器的时间

输出：

无

1. 函数 time\_sleep

用途：休眠

输入：

time\_value，int，休眠的时间

输出：

无

1. 文件名： sys\_inter\_act.py

用途：和系统交互（shell命令相关）

函数：

1. 函数 exe\_sys\_cmd\_get\_echo

用途：执行系统shell命令，获取回显

输入：

os\_cmd，str，shell命令

输出：

str，回显信息

1. 函数 exe\_sys\_cmd

用途：执行系统shell命令

输入：

os\_cmd，str，shell命令

输出：

无

1. 函数 exe\_sys\_cmd\_get\_echo\_v3

用途：执行系统shell命令，获取回显和错误码

输入：

os\_cmd，str，shell命令

输出：

cmd\_echo，str，回显信息

echo\_error\_code，int，错误码

1. 函数 exe\_sys\_cmd\_v3

用途：执行系统shell命令，获取执行结果

输入：

os\_cmd，str，shell命令

输出：

bool，成功/失败

1. 函数 split\_sys\_cmd\_echo

用途：从shell执行后的回显中，提取所需的列

输入：

os\_cmd\_echo，str，shell命令的回显

position，int，需要获取的列（编号从0开始）

echo\_line\_split\_len，int，回显内容每行最大拆分的列数量

输出：

os\_cmd\_echo\_split，list，抽取回显中的某一列数据，形成一个list

1. 函数 get\_service\_pid

用途：根据服务名（或相关正则）获取进程的pid列表

输入：

service\_name，str，服务名（或相关正则）

输出：

pid\_list，list，获取到的进程的pid的list

1. 函数 get\_service\_path\_by\_pid

用途：根据进程的pid列表获取进程所在的绝对路径

输入：

pid\_list，list，进程的pid列表

输出：

pid\_path\_dict，dict，{“pid”:”proc\_path”,…}

1. 函数 monitor\_port\_by\_pid

用途：根据进程的pid和期望的监听端口，判断服务是否监听正常

输入：

pid，str，进程的pid

listen\_port，str，期望的监听端口

输出：

bool，监听正常/失败

1. 函数 start\_service

用途：启动本地服务，并检测是否启动正常

输入：

service\_name，str，服务的名字（或相关的正则）

service\_path，str，服务归属的路径（很可能是启动脚本所在的路径）

start\_script，str，启动脚本的绝对路径和文件名称

listen\_port，服务启动后期望的监听端口

time\_out，监测的超时时间

输出：

is\_ok，bool，服务启动成功/失败

service\_pid，str，服务的进程的pid

start\_time\_cost，float，服务启动所花的时间

1. 函数 stop\_service

用途：停止本地服务，并检测是否停止成功

输入：

service\_pid，str，服务的进程的pid

time\_out，监测的超时时间，监测时间内，三次kill停止服务不成功则强制杀死进程

输出：

is\_ok，bool，服务停止成功/失败

1. 函数 stop\_all\_service\_forcibly

用途：强制停止本地服务

输入：

service\_name，str，服务的名字（或相关的正则）

time\_out，监测的超时时间

输出：

is\_ok，bool，服务停止成功/失败

# proc库说明：

1. 文件名： package\_deploy.py

用途：部署服务到指定的线下服务器，包括下载源码，编译，打包，修改配置，下发到被测服务器，修改测试数据文件，启动服务器等一系列流程。

函数：

1. 函数 get\_job\_ini

用途：rsync方式获取线上的 job.ini文件，用于获取其中的版本信息和yum依赖信息

输入：

server，str，线上服务器地址

path，str，线上服务器上的job.ini所在的路基

local\_path，str，本地的目录

输出：

bool，成功/失败

1. 函数 modify\_job\_ini

用途：根据测试需求，修改job.ini文件

输入：

config\_modify\_list，来自配置文件(config\_modify\_XXX.json)

local\_path，str，job.ini所在的路径

输出：

bool，成功/失败

1. 函数 svn\_get\_package

用途：根据job.ini文件，下载源码

输入：

job\_ini，str，job.ini路径

local\_path，str，本地路径，用于存放下载的代码

svn\_user，str，svn的用户，置空则用通用账户

svn\_passwd，str，svn的密码，置空则用通用账户

输出：

str，版本的tag

1. 函数 get\_conf

用途：用rsync方式从线上获取cfg文件，用于线下测试的template

输入：

server，str，线上服务器地址

conf\_online，str，线上服务器上的cfg的绝对路径

local\_path，str，本地路径，用于存放下载的代码

config\_modify\_list，来自配置文件(config\_modify\_XXX.json)

输出：

bool，成功/失败

1. 函数 gen\_conf

用途：根据测试需求，修改cfg文件

输入：

config\_modify\_list，来自配置文件(config\_modify\_XXX.json)

local\_path，str，本地路径，用于存放下载的代码

输出：

bool，成功/失败

1. 函数 compile\_dailybuild

用途：生成测试代码的dailybuild二进制文件，比如VRQO

输入：

job\_ini，str，job.ini路径

local\_path，str，本地路径，用于存放下载的代码

输出：

bool，成功/失败

1. 函数 distribute\_dailybuild

用途：分发dailybuild和配置文件到多台被测服务器

输入：

server\_info\_list，线下服务器的信息，来自配置文件conf\_XXX.json的"offline\_server\_info"

local\_path，str，本地路径，用于存放下载的代码

remote\_path，str，线下服务器的部署路径，为空的话则取配置文件conf\_XXX.json的"offline\_server\_info"

remote\_data\_path，str，线下服务器的测试数据所在路径

build\_tag，str，build标记，用于追加在remote\_path后面，以区分不同的构建

输出：

bool，成功/失败

1. 函数 distribute\_customized\_data

用途：分发构建测试数据的脚本和配置文件到多台被测服务器，并构建测试数据

输入：

server\_info\_list，线下服务器的信息，来自配置文件conf\_XXX.json的"offline\_server\_info"

offline\_data\_info，来自配置文件conf\_XXX.json的"offline\_data\_info"，用于指明需要传到被测机器的脚本和数据修改的配置文件。

remote\_path，str，线下服务器的部署路径，为空的话则取配置文件conf\_XXX.json的"offline\_server\_info"

build\_tag，str，build标记，用于追加在remote\_path后面，以区分不同的构建

输出：

bool，成功/失败

1. 函数 start\_module\_service

用途：启动被测服务器上的被测模块

输入：

server\_info\_list，线下服务器的信息，来自配置文件conf\_XXX.json的"offline\_server\_info"

remote\_path，str，线下服务器的部署路径，为空的话则取配置文件conf\_XXX.json的"offline\_server\_info"

module\_name，str，被测服务的可执行程序的文件名

conf\_name，str，被测服务的启动所需的配置文件名

build\_tag，str，build标记，以区分不同的构建

输出：

bool，成功/失败

1. 函数 processor

用途：处理函数的总入口

输入：

deploy\_conf，dict，配置文件conf\_XXX.json解析后的内容

build\_tag，str，build标记，以区分不同的构建

输出：

无

# 配置文件说明：

1. 文件名： conf\_template.json

用途：build所必须的信息，使用前请构造自己的配置文件。

{

"service\_info":{ ---被测服务的可执行文件名字和类型（类型同名字即可）

"service\_name":"vrqo",

"service\_type":"vrqo"

},

"deploy\_info":

{

"deploy\_local\_path":"/search/odin/tester/vrqo/" ---中控机上的本地路径，同代码中的local\_path

},

"svn\_info":{ ---svn的信息，用于下载被测服务的源码

"username":"qa\_svnreader",

"password":"'New$oGou4U!'",

"svn\_link": "http://svn.sogou-inc.com/svn/websearch4/VH/vr\_QO/branches/opt\_vrqo.20191014"

},

"online\_info":{ ---线上机器的信息，用户获取job.ini和配置文件

"module\_machine":"rsync.vrqo01.web.1.djt.ted",

"module\_path":"odin/search/odin/daemon/vrqo/",

"module\_conf":"odin/search/odin/daemon/vrqo/conf/vrqo\_1djt.cfg"

},

"offline\_server\_info":[ ---线下被测服务器的信息，用于部署测试版本

{

"addr":"rsync.offvrqo01.in.sjs.ted",

"username":"root",

"password":"noSafeNoWork@2014",

"deploy\_path":"/search/odin/test/"

},

{

"addr":"rsync.offvrqo02.in.sjs.ted",

"username":"root",

"password":"noSafeNoWork@2014",

"deploy\_path":"/search/odin/test/"

},

{

"addr":"rsync.offvrqo03.in.sjs.ted",

"username":"root",

"password":"noSafeNoWork@2014",

"deploy\_path":"/search/odin/test/"

},

{

"addr":"rsync.offvrqo04.in.sjs.ted",

"username":"root",

"password":"noSafeNoWork@2014",

"deploy\_path":"/search/odin/test/"

}

],

"offline\_data\_info":{ ---线下被测服务器的数据部署信息，用于部署数据构建工具和对应的配置文件，然后再被测机器上构建测试数据

"data\_base\_path":"/search/data/vrqo\_data/",

"data\_modify\_path":"",

"data\_script":"./proc/vrqo\_data\_deploy.py",

"data\_conf":"./conf/data\_modify\_template.json"

},

"config\_modify\_file":"./conf/config\_modify\_template.json" ---用于修改线下被测服务的配置文件

}

1. 文件名： config\_modify\_template.json

用途：用于指示如何修改被测服务的配置文件，以及修改job.ini文件，使用前请构造自己的配置文件。

片段截取如下：

[

{

"config\_file": "job.ini",

"config\_type": "ini",

"modContent":[ ---修改job.ini文件

{

"anchor": ["main","build\_requires"], ---修改点锚定

"action": "add", ---修改点方式，支持add，delete，replace

"content": "pcre2 pcre2-devel" ---修改后的内容

},

{

"anchor": ["main.svn","."],

"action": "replace",

"content": "http://svn.sogou-inc.com/svn/websearch4/VH/vr\_QO/branches/opt\_vrqo.20191014"

}

]

},

{

"config\_file": "vrqo.cfg.in.func",

"config\_type": "cfg",

"enable": true,

"modContent":[

{

"anchor": ["isCacheExternal"], ---修改点锚定

"action": "replace", ---修改点方式，支持add，replace

"content": "no", ---修改后的内容

},

1. 文件名： data\_modify\_template.json

用途：用于指示如何修改被测服务的测试所需的测试数据，使用前请构造自己的配置文件。

片段截取如下：

{

"data\_base\_path":"/search/data/vrqo\_data/",

"data\_modify\_path":"",

"dev\_data": [ --开发提供的新增的测试数据所在的服务器信息

{

"addr":"10.134.104.61",

"username":"root",

"password":"noSafeNoWork@2014",

"data\_path":["/search/odin/xujiang/vrqo\_data/base/external\_conf/"]

}

],

"pre\_server\_info": { --线上预发机器的信息，用于获取线上测试数据

"addr":"10.134.64.32",

"username":"root",

"password":"vrqo@2019"

},

"vrqo\_VRInfo": {

"class\_ids": ["20005005","70041102","21165501"]

},

"vrqo\_QOConfig": {

"class\_ids": ["20005005","70041102","21165501"]

},

"vr\_files": {

"class\_ids": ["20005005","70041102","21165501"]

},

"vrqo\_TransVrType": [

{

"anchor": "$", ---修改点锚定

"action": "append", ---修改点方式，支持add，replace，append，insert，delete

"content": "test\tWIRELESS.SELFMANAGED.runpingnianpanding" ---修改后的内容

}

]

}