

A Generalized Swagger (OpenAPI) Centered Web Services Testing Framework

Article Format Of Presentation

Document #PLPC-180057

Version 0.1

October 23, 2018

This Document is Available on-line at:
<http://www.by-star.net/PLPC/180057>

Neda Communications, Inc.
Email: <http://www.by-star.net/contact>

Copyright © 2014 Neda Communications, Inc.

Permission is granted to make and distribute complete (not partial) verbatim copies of this document provided that the copyright notice and this permission notice are preserved on all copies.

Contents

I Overview Of mmwsIcm Web Services Testing Framework	2
1 Invokers Development Model	2
1.1 Invoker ICMs Development Model	2
1.2 Invoker Framework: Ingredients	2
II rinvoker	4
2 rinvoker.py	4
2.1 rinvoker.py Overview	4
III Operation Scenarios	5
3 Invoke-Specifications, Invoke-Verification, Invoke-Reporting	5
3.1 Model Of Invoke – Specification, Verification And Reporting – Scenarios	5
3.2 Scenario Specification For Sequences Of Invocations	5
4 Invoker-Apps Development Model	6
4.1 Invoker-Apps Can Easily Build On unisos.mmwsIcm Capabilities	6
5 Benefits And Advantages Of The Generalized Swagger Centered Invocation Model	7
5.1 Taking Full Advantage Of Service Specification For Testing And Development	7

List of Figures

Part I

Overview Of mmwsIcm Web Services Testing Framework

Outline of Part I – Overview Of mmwsIcm Web Services Testing Framework

Contents

Notes:

1 Invokers Development Model

1.1 Invoker ICMs Development Model

Given a Service Definition (a swagger file) and a Performer Server, you should be able to conveniently Invoke any of the offered Operations through:

1. swagger.ui interface – usually offered by the Performer Server
 2. unisos.mmwsIcm :: rinvoke.py – command line and batch oriented equivalent of swagger.ui
 3. unisos.mmwsIcm :: ro.py – invoke-specification – invoke-verification – invoke-reporting
 4. unisos.mmwsIcm – rich library for building invoker Apps
-

Notes:

1.2 Invoker Framework: Ingredients

Main software packages that implement the framework include:

1. Python Bravado – Equivalent of Invoker Codegenartor But Better
2. unisos.icm – Interactive Command Module
 - Makes icm.Cmnd classes invocable at command-line

- do-icm :: Direct Operation ICMs (Used by performers)

3. unisos.mmwsIcm

- unisos.mmwsIcm.wsInvoker.py – Maps invocations to http requests
- unisos.mmwsIcm.ro.py – Abstracts invoke-specifications
- unisos.mmwsIcm.rinvoker.py – Maps command-line args to invocations

Try It: `pip install unisos.mmwsIcm`

Notes:

Part II

rinvoker

Outline of Part II – rinvoker

Contents

Notes:

2 rinvoker.py

2.1 rinvoker.py Overview

Allows you to list all possible invocations based on a service specification (swagger file).

```
rinvoker.py --svcSpec="http://petstore.swagger.io/v2/swagger.json" -i svcOpsList
```

Allows you to fully specify an invocation on command line. Example:

```
rinvoker.py --svcSpec="http://petstore.swagger.io/v2/swagger.json"
  --resource="user" --opName="createUser" -i rinvoker
  bodyStr="{...}"
```

Notes:

Part III

Operation Scenarios

Outline of Part III – Operation Scenarios

Contents

Notes:

3 Invoke-Specifications, Invoke-Verification, Invoke-Reporting

3.1 Model Of Invoke – Specification, Verification And Reporting – Scenarios

- Invoke Scenarios Are pure python specification of sequence of invocations.
 - Invoke-Expect Scenarios Are pure python specification of sequence of invocations subject to preparations and post-invoke verification and reporting.
 - opInvoke class allows for complete invoke specification and complete results to be fully captured.
-

Notes:

3.2 Scenario Specification For Sequences Of Invocations

In pure python specify invocation of each operation, for example:

```
rosList.opAdd(  
    svcSpec=akSkToken_svcSpec,    # e.g. "../SvcSpec/akSkToken.json"  
    perfSap=akSkToken_perfSap,    # e.g. "http://localhost:8080/access/v1.0"  
    resource="token",  
    opName="obtain",  
    roParams=ro.Ro_Params(  
        headerParams=None,  
        urlParams=None,  
        bodyParams={
```

```
        'accessKey': "someOtherId",
        'secretKey': "someOtherCredentials",},),
    roResults=None,
)
```

Then load that list of invocations and subject them to sequential invocations.

Notes:

Building on the previously mentioned Operation Specification, in pure python you can the specify Operation Expectations, for example:

```
roExpectationsList.opExpectationAdd(
    # Above mentioned operation specification
    # ...
    preInvokeCallables=[],
    postInvokeCallables=[ verify_akSkTokenObtainRo, ],
)
```

For example, `preInvokeCallables(ro.Ro_OpExpectation)` can include a function that initializes the DB.

For example, `postInvokeCallables(ro.Ro_OpExpectation)` can include a function that verifies the operation's result was as expected and then reports success or failure.

Notes:

4 Invoker-Apps Development Model

4.1 Invoker-Apps Can Easily Build On `unisos.mmwsIcm` Capabilities

- Bravado does invoker code-generation on the fly.
- `unisos.mmwsIcm.opInvoke` – Abstracts invoke-specifications
- `unisos.mmwsIcm.wsInvoker` – Allows for invocation and verification of `opInvoke`

With these in place, building Invoke-Apps becomes very simple.

Notes:

5 Benefits And Advantages Of The Generalized Swagger Centered Invocation Model

5.1 Taking Full Advantage Of Service Specification For Testing And Development

- The Generalized Model and Capabilities Presented Here Apply To Any Service That Exposes Its Swagger Specifications
- A great deal of automation capabilities have become possible based on swagger specifications.
- The Testing Framework Of (invoke-specification, invoke-verification and invoke-reporting) permits for disciplined complete external testing of web-services based on swagger specifications.

Very Often, These Best Current Practices Are Not Being Followed.

Notes:

References