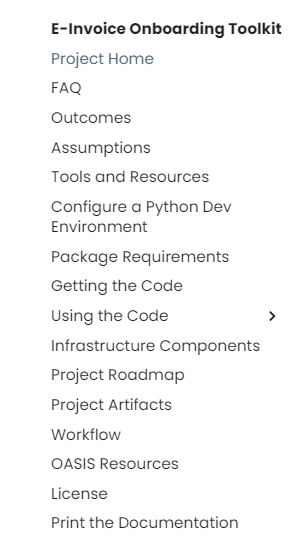
E-invoice Onboarding Toolkit

Repository

https://github.com/BPC-OpenSourceTools/e-Invoice-Onboarding-Toolki

[t](https://github.com/BPC-OpenSourceTools/e-Invoice-Onboarding-Toolkit/)

[/](https://github.com/BPC-OpenSourceTools/e-Invoice-Onboarding-Toolkit/)



# Table of Contents

1. [**Project Home**](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#index-project-home)

!"[1.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "index-welcome) [Welcome](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "index-welcome) !"[1.2](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "index-this-project) [This Project](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "index-this-project)

1. [**Questions...**](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#faq-questions)
2. [**Project Outcomes:**](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#outcomes-project-outcomes)

!"[3.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#outcomes-functionality) [Functionality](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#outcomes-functionality)

!"[3.2](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#outcomes-implementation) [Implementation](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#outcomes-implementation)

!"[3.3](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#outcomes-to-do) [To do](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#outcomes-to-do)

!"[3.4](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#outcomes-notes) [Notes](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#outcomes-notes)

1. **[E-invoice Onboarding Toolkit](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "assumptions-diving-in-to-the-e-invoice-onboarding-toolkit) [5](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "tools_and_resources-tools-and-resources)** [**Tools and Resources**](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#tools_and_resources-tools-and-resources)

!"[5.0.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#tools_and_resources-quick-guide) [Quick Guide](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#tools_and_resources-quick-guide)

!"[5.0.2](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#tools_and_resources-programming-languages) [Programming Languages](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#tools_and_resources-programming-languages)

!"[5.0.3](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#tools_and_resources-computer-spec) [Computer Spec](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#tools_and_resources-computer-spec)

!"[5.0.4](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#tools_and_resources-operating-system) [Operating System](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#tools_and_resources-operating-system)

1. **[Configure a Python Development Environment](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "python_dev_env-configure-a-python-dev-environment)**

!"[6.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#python_dev_env-installing-python) [Installing Python](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#python_dev_env-installing-python)

!"[6.2](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#python_dev_env-create-a-virtual-environment-to-use-for-dev) [Create a virtual environment to use for Dev](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#python_dev_env-create-a-virtual-environment-to-use-for-dev)

1. [**Project Package Requirements**](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#requirements-project-package-requirements)

!"[7.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#requirements-currently-implemented-packages) [Currently implemented packages:](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#requirements-currently-implemented-packages) [**8**](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#working_with_the_code-getting-the-code) **[Getting the code](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "working_with_the_code-getting-the-code)**

!"[8.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#working_with_the_code-pulling-the-code-from-github) [Pulling the code from GitHub](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "working_with_the_code-pulling-the-code-from-github)

I Using the Code

1. [**Integrating Code Modules**](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#using_the_modules-integrating-code-modules)

!"[9.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#using_the_modules-the-package-structure-and-using-the-modules) [The Package Structure and Using the Modules](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#using_the_modules-the-package-structure-and-using-the-modules)

!"[9.2](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#using_the_modules-additional-files) [Additional Files](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#using_the_modules-additional-files)

!"[9.3](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#using_the_modules-note-on-classes-with-modules) [Note on classes with modules.](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#using_the_modules-note-on-classes-with-modules)

1. **[E-invoice Onboarding Toolkit](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "test_cases-e-invoice-onboarding-tool-kit)**

!"[10.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#test_cases-working-with-the-code) [Working with the code](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#test_cases-working-with-the-code) !"[10.1.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#test_cases-test-cases) [Test Cases](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#test_cases-test-cases)

1. **[Integration Start-to-Finish](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "start_to_finish-start-to-finish)**

!"[11.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#start_to_finish-start-to-finish-integration) [Start-to-Finish Integration](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "start_to_finish-start-to-finish-integration)

1. **[Jupyter Notebooks on Google Colab](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "google_colab_pages-jupyter-notebooks-on-google-colab)**

!"[12.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#google_colab_pages-colab-sandboxes) [Colab Sandboxes](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "google_colab_pages-colab-sandboxes)

!"[12.1.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#google_colab_pages-urn-hashing-and-dns-naptr-lookup) [urn hashing and DNS NAPTR lookup.](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "google_colab_pages-urn-hashing-and-dns-naptr-lookup)

!"[12.2](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#google_colab_pages-smp-query) [SMP query](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#google_colab_pages-smp-query)

!"[12.3](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#google_colab_pages-ebms-message-header-validation) [ebMS Message Header validation](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#google_colab_pages-ebms-message-header-validation)

1. **[Additional Infrastructure Build-out](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "infrastructure_components-additional-infrastructure-build-out)**

!"[13.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#infrastructure_components-sml) [SML](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#infrastructure_components-sml)

!"[13.2](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#infrastructure_components-smp) [SMP](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#infrastructure_components-smp)

1. [**Project Roadmap**](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#project_roadmap-project-roadmap)

!"[14.0.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#project_roadmap-discovery-1-sml-naptr-dns-lookup) [Discovery #1 - SML NAPTR DNS Lookup](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#project_roadmap-discovery-1-sml-naptr-dns-lookup)

!"[14.0.2](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#project_roadmap-discovery-2-smp-rest-api-query) [Discovery #2 - SMP REST API Query](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#project_roadmap-discovery-2-smp-rest-api-query)

!"[14.0.3](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#project_roadmap-delivery-as4-message-exchange) [Delivery - AS4 Message Exchange](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#project_roadmap-delivery-as4-message-exchange)

1. **[The Repo(sitory)](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "artifacts-the-repository)**

!"[15.0.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#artifacts-repository-layout) [Repository Layout](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#artifacts-repository-layout)

1. **[Sample git workflow](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "git_workflow-sample-git-workflow)**

!"[16.0.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#git_workflow-a-minimal-git-script-to-work-with-the-code) [A minimal git "script" to work with the code.](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#git_workflow-a-minimal-git-script-to-work-with-the-code)

1. [**OASIS Resources**](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#oasis_documentation-oasis-resources)

!"[17.1](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#oasis_documentation-documents) [Documents](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#oasis_documentation-documents)

!"[17.2](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#oasis_documentation-xml-schema-definitions) [XML Schema Definitions](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#oasis_documentation-xml-schema-definitions) !"[17.3](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#oasis_documentation-namespace-uris) [Namespace URIs](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#oasis_documentation-namespace-uris)

1. [**MIT License**](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#_license-mit-license)

# 1 Project Home

## 1.1 Welcome

**Welcome to the E-invoice Onboarding Toolkit**

This is a repository for open source software tools created to facilitate market adoption of e-invoices implemented conformant with the Four-Corner interoperability model framework.

1

.

Hashing functionality to derive the urn for look-up in a DNS NAPTR record.

2

.

DNS NATPR lookup and extract the relevant SMP URI.

3

.

Two REST requests to an SMP server to retrieve a Corner 3 URI.

4

.

Execute the REST requests to the SMP server.

5

.

Extract the Corner 3 endpoint URI from the response from the SMP server.

6

.

Validate an e-Invoice ebMS message header for compliance with an AS4 conformance profile.

**Outcomes**



For information about e-invoices and implementing the Four-Corner Model please visit the [Business Payments Coalition website](https://businesspaymentscoalition.org/electronic-invoices).

Additional documentation, reference materials, and standards can be found on the [Oasis-Open.org website](https://www.oasis-open.org/). Start with the [ebXML specification](http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/os/ebms_core-3.0-spec-os.html)

## 1.2 This Project

!"[Project Home](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#.)

!"[FAQ](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#faq)

!"[Outcomes](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#outcomes)

!"[Assumptions](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#assumptions)

!"[Tools and Resources](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#tools_and_resources)

!"[Configure a Python Environment](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#python_dev_env)

!"[Package Requirements](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#requirements)

!"[Getting the Code](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#working_with_the_code)

!"Using the Code

!"[Using the Modules](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#using_the_modules)

!"[Test Cases](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#test_cases)

!"[Start-to-Finish example](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#start_to_finish)

!"[JupyterLab/Notebooks](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/" \l "google_colab_pages)

!"[Infrastructure Components](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#infrastructure_components)

!"[Project Roadmap](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#project_roadmap)

!"[Project Artifacts](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#artifacts)

!"[Workflow](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#git_workflow)

!"[Oasis Resources](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#oasis_documentation)

!"[License](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#_license)

### 1.2.0.1 No Representations or Warranties

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

2 Questions...

**Q:**

Who is the audience for this project?

**A:**

This software is intended for those interested in participating as service endpoints in a Four-Corner Model framework. The code to implement in the repository is written in the Python programing language. Other toolsets to facilitate the initiative, such as Markdown or Docker, may also be incorporated where appropriate.

**Q:**

What do I need in order to use this code?

**A:**

**Python**

1. Some knowledge of Python
2. Python 3.6 or greater; *Python 3.10 or greater is recommended.*

**Q:**

Does this code provide a full end-to-end solution to process an e-invoice?

**A:**

This code answers some very domain specific questions regarding core functionality supporting discovery and delivery components in a Four-Corners exchange framework.

Specifically it's helpful with:

1. Discovery:
   1. The hashing functionality to derive the URN for look-up in a DNS NAPTR record.
   2. How to do the DNS NATPR lookup and extract the relevant SMP URI.
   3. How to construct the two REST requests to an SMP server to retrieve a Corner 3 URI.
   4. How to execute the REST requests to the SMP server.
   5. How to extract the Corner 3 endpoint URI from the response from the SMP server.
2. Delivery:
   1. Validating an e-Invoice ebXML message header for compliance with an AS4 conformance profile.

**Q:**

How do I use the code?

**A:**

There are a number of ways the code can be examined or worked with:

1. [Start to finish example of the URI discovery process](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#start_to_finish).
2. [Test Cases](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#test_cases) which demonstrate functionality of the modules.
3. Jupyter Notebook sandbox environments at [Google Colab Pages](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#google_colab_pages) which isolate and demonstrate the code in a sandbox.
4. Package/Library API see the [Index](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#.) for links to the code API on the modules themselves.
5. Integration of an example discovery process implemented in a [Flask application on a Docker container](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#faq-flask_integration_on_docker.md) (forthcoming).

### 2.0.0.1 No Representations or Warranties

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

3 Project Outcomes:

## 3.1 Functionality

1. Discovery:
   1. Hashing functionality to derive the URN for look-up in a DNS NAPTR record.
   2. DNS NATPR lookup and extract the relevant SMP URI.
   3. Two REST requests to an SMP server to retrieve a Corner 3 URI.
   4. Execute the REST requests to the SMP server.
   5. Extract the Corner 3 endpoint URI from the response from the SMP server.
2. Delivery:
   1. Validate an e-Invoice ebXML message header for compliance with an AS4 conformance profile.

## 3.2 Implementation

1. Functional Python code:

Construct the URN from the specification, schema ID, and party ID, urn\_hasher.py

Hash the URN per the requirements to create a NAPTR record for a DNS look-up to obtain SMP service URI.

urn\_hasher.py accessor.py

Do the DNS look-up to obtain the SMP service URI.

accessor.py dns\_query.py

Query the SMP URI using the ebXML spec.

accessor.py smp\_query.py

Dataclass object comprised of specification, schema ID, party ID, and a sample JSON e-Invoice payload.

einvoice\_message\_package.py line\_item\_py party\_address.py semnantic\_model.py urn.py

Test cases implemented using Test Driven Development test\_accessor.py test\_app\_logging.py

test\_create\_sample\_data.py test\_create\_tracking\_id.py test\_dns\_query.py test\_import\_xsd.py test\_line\_item.py test\_party\_address.py test\_semantic\_model.py test\_smp\_query.py test\_urn.py test\_urn\_hasher.py

Logging implemented through custom logging using standard Python modules.

app\_loggiing.py

2. Documentation and code artifacts for Infrastructure components:

Demonstrating implementation of DNS infrastructure using Amazon Route53 and code to provision, update, and delete NATPR records, acting as the SML.

Demonstrating implementation of SMP infrastructure to reply to the REST API for service functionally and AS4 final endpoint. 3. Documentation

Previously created documentation and diagrams which were outcomes of analysis and process review.

Jupyter Lab Notebooks running on Google Colab for real-time examples of a development sandbox.

Hash URN and SML query Lab Notebook (Hash URN and SML query are in the same Notebook)

SMP query Lab Notebook

Review of AS4 XSD spec Lab Notebook

Validation of ebMS sample message header against AS4 XSD Lab Notebook

~~Implementation Guide~~ Documentation site written by SEs and prepared with BSAs to facilitate utilization of the code and work product to be delivered ~~via readthedocs.org site~~ though the github repository.

~~readthedocs.org site registration/creation~~ Create document set using **mkdocs-material** hosted on github repository.

Create ~~list of assumptions~~ **FAQ** and **Outcomes** pages for starting point/baseline documentation.

Aggregation of above referenced artifacts hosted on github.

## 3.3 To do

1. A deployment package which includes Python code to:

(In-progress)\_ Creation of "final" Python package which delivers code artifacts as a library.

1. Documentation

(In-progress) Generated from python docstring using ~~Sphinx~~ **mkdocs-material** framework

(In-progress) Documentation of supporting infrastructure including DNS and SMP provisioning specifically called out as a parallel value-add result of the project.

## 3.4 Notes

Test-driven development methodology is being implemented to include test cases for code as it is being developed and delivered.

CI/CD process implemented via Github workflow has been validated to ensure PEP8 code standards and checks using Flake8, pylint, and pytest are valid. All changes and updates to code must pass CI/CD before it's merged into the repo.

### 3.4.0.1 No Representations or Warranties

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 4 E-invoice Onboarding Toolkit

**4.0.0.1 About**

This project offers examples of Python code written to interact with an access point or discovery service of a Four-Corner Model of an e-Delivery Network.

Please see the [Business Payments Coalition](https://businesspaymentscoalition.org/electronic-invoices/) website for more information and an explanation of the Four-Corner Model.

**4.0.0.2 Assumptions**

**4.0.0.2.1 THE FOUR-CORNER MODEL**

The software included with the project assumes a baseline understanding of the Four-Corner Model and its constituent components.

**4.0.0.2.2 PYTHON**

The primary programming language used is Python. To implement and run the code in [this project](https://github.com/BPC-OpenSourceTools/e-Invoice-Onboarding-Toolkit) requires a working knowledge of Python. This wiki and additional documentation are intended to further outline how the software is designed to implement the Four-Corner Model.

## **4.0.0.3 No Representations or Warranties**

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 5 Tools and Resources

**GOAL:** Create, test, deploy, and maintain code to the highest professional standards.

**HOW:** Tools and best practices which facilitate development of high quality code with testable and reproducible outcomes.

## 5.0.1 Quick Guide

**Tool**

**Minimal requirements**

Programming Language

Python 3.6 or above.

Computer

Supports running Python 3.6 or above.

OS

Mac, Windows, or Windows w/WSL2.

Documents and resources.

GitHub and

BPC-Technical-Workgroup-Folder - Google Dri

[v](https://na01.safelinks.protection.outlook.com/?url=https%3A%2F%2Fdrive.google.com%2Fdrive%2Ffolders%2F15aJogTieUuqukjDHm5AVQVVm-PVukPs9&data=04%7C01%7C%7C0ce76ff21ff048af0c2b08d8ef9790ff%7C84df9e7fe9f640afb435aaaaaaaaaaaa%7C1%7C0%7C637522781535891404%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C1000&sdata=l1%2BAoHrIHr8przKXkn6pSCQTGnGXKMcOjQnzY0jpqws%3D&reserved=0)

[e](https://na01.safelinks.protection.outlook.com/?url=https%3A%2F%2Fdrive.google.com%2Fdrive%2Ffolders%2F15aJogTieUuqukjDHm5AVQVVm-PVukPs9&data=04%7C01%7C%7C0ce76ff21ff048af0c2b08d8ef9790ff%7C84df9e7fe9f640afb435aaaaaaaaaaaa%7C1%7C0%7C637522781535891404%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C1000&sdata=l1%2BAoHrIHr8przKXkn6pSCQTGnGXKMcOjQnzY0jpqws%3D&reserved=0)

## 5.0.2 Programming Languages

The primary programming language used is Python. Knowledge of other enabling technologies, specifically shell scripting (e.g., ZSH, BASH, or PowerShell) and CommonMark or GitHub Flavored Markdown may be helpful.

All effort will be made to remain within the [Python Standard Library](https://docs.python.org/3.10/library/index.html). Other publicly hosted packages with an open source license may be implemented.

**5.0.2.1 Version of Python**

*Select a minimum version of Python 3.6. This will include newer features such as f-string.* 



## 5.0.3 Computer Spec

**You will need a computer which is able to download and run Python 3.6.**



## 5.0.4 Operating System

**OS**

**Considerations**

Mac OS

Included system version of Python 2.x must be respected. Make sure to use Python virtual

environments. Homebrew package manager is a plus.

Windows

Latest version of Python are now available on the Windows Store. Anaconda3 is an especially good

option in this OS.

**OS**

**Considerations**

Windows

w/WSL2

Great implementation of Ubuntu on Windows is truly impressive. Integration of virtual

environments and Python executable with an IDE like VS Code can be finicky.

Linux

Native support for distributed technologies, i.e., Kubernetes and Docker.

Chrome OS

Limited on-device resources but a growing number of online and cloud development options, e.g.,

AWS, Azure, OpenShift, JupyterLab and Notebooks.

Raspberry PI

With native Python support, Linux packages, and cloud options there is no reason development is not an

option.

These options show that there is nothing proprietary, exclusive, or given preference other than resource availability and personal preference.

**5.0.4.1 Additional Configuration and Environment Considerations**

Future looking consideration for implementation of additional Python enabling technology in support of scalability, portability, and resiliency includes:

**5.0.4.1.1 FRAMEWORKS:**

!"Django !"Flask

!"FastAPI

!"OpenAPI

**5.0.4.1.2 CONTAINERIZATION AND CLOUD SERVICES:**

!"Docker

Local testing of Docker containerization may be done on a desktop. However, a cloud implementation is the typical endpoint for a container deployment.

Cloud services such as:

!"Amazon Web Services (AWS) including lambdas and Route53 DNS

!"Microsoft Azure

!"RedHat OpenShift

Additional infrastructure components may be implemented by the Project for testing or prototyping utilization of cloud services.

The implementation of a cloud service and choice of provider is entirely the responsibility of the implementer of the code. Code artifacts included here which implement cloud services are examples for research and educational purposes only. No preference or endorsement is given to any provider.

**5.0.4.2 Python Programming Methodologies, Standards, and Tools:**

Some standards:

!"[PEP20](https://pep20.org/)

!"[PEP8](https://pep8.org/)

!"[Guiding Design Principles](https://nsls-ii.github.io/scientific-python-cookiecutter/guiding-design-principles.html) from the [Scientific Python Cookiecutter](https://nsls-ii.github.io/scientific-python-cookiecutter/index.html).

!"[Python Doc](https://www.python.org/doc/) the official Python web site page of references to more documentation.

!"*See the list of books below for additional guidance on standards and best practices in Python development*.

Tools to implement the standards(preferred ones in **bold**) .

!"**flake8**

!"autopep8

!"**pylint**

!"**pytest**

!"unittest

!"black (Careful! - Handles most formatting well but line character length may not be handed as desired.)

!"bandit

!"**mypy**

!"**pycodestyle**

!"**pydocstyle**

Development methodologies \* Documentation and use of  \* Domain Driven Design \* Test Driven Development \* Agile principles applied appropriate to the size and state of the project.

**5.0.4.3 Books**

There are many great reference materials in print and on the Internet about Python development.

Below are references that may be helpful with Python.

!"The Hitchhiker's Guide to Python by Kenneth Reitz and Tanya Schlusser. On-line for free at [docs.python-guide.org](https://docs.python-guide.org/).

!"Serious Python by Julien Danjou

!"Domain-Driven Design Distilled by Vaughn Vernon

!"Domain Driven Design: Tackling Complexity in the Heart of Software by Eric Evans

!"Test Driven Development with Python: Obey the Testing Goat, etc. by Harry J.W. Percival. Also [available online for free](https://www.obeythetestinggoat.com/).

!"Architecture Patterns with Python by Harry J.W. Percival and Bob Gregory

!"Pro Git by Scott Chacon and Ben Straub. [Available as a free download](https://git-scm.com/book/en/v2) under an open source license.

### 5.0.4.4 No Representations or Warranties

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 6 Configure a Python Dev Environment

Respecting individual preferences and work style below are some suggested guidelines for creation of a Python development environment. These are consistent with Python standards and best practices and appropriate as a starting point for professional software development in Python.

## 6.1 Installing Python

The correct method of installing Python varies depending on the OS. Here are some considerations based on popular OS.

### OS Considerations

MacOS Mac OS comes configured with Python 2.x. Don't attempt to remove it or replace it . [Homebrew](https://brew.sh/) is a

package installer for Mac. However given some past and outstanding issues between Homebrew and Python, versions of Python prior to Python 3.7 may generate errors on install. [Apple XCode Developer Tools](https://developer.apple.com/xcode/) installs Git and a version of Python. Available through the App Store it is a large download and system intensive program. A more granular installation would be the [XCode](https://mac.install.guide/commandlinetools/4.html)



[Command Line Tools](https://mac.install.guide/commandlinetools/4.html) which installs Git combined with a Python install pulled directly from Python.org.

WindowsOS Python is now available on the [Windows Store](https://www.microsoft.com/en-us/store/apps/windows), though updates may lag behind current Python releases. Ease of setting environment variables makes it easy to maintain multiple installed versions simultaneously. This configuration is the easiest to integrate [VS Code](https://code.visualstudio.com/) with Python.

Windows A fully native Python install on Ubuntu is available for [Windows Subsystem for Linux 2](https://docs.microsoft.com/en-us/windows/wsl/install). The version of

WLS2 Python may need to be updated from a secondary repository as the official Ubuntu version trails official Python releases. Integration between VS Code running on Windows and integration with WSL2 Python may not be seamless.

Linux A variety of package managers based on the installed distro may impact the ease of installing current Python releases and updates, but overall this possibly the easiest to "OS" to install and maintain Python.

### Python3

Python is "officially" referred to and invoked by specifying either Python 2.x as "Python2" or Python 3.x.x as "Python3." The difference is generally trivial except on Macs where Python2 is included as part of the OS install.

On Mac and Linux based systems, adding an alias to .bashrc or .zshrc is an easy way to prevent inadvertent references to an incorrect version of Python, e.g.,

alias python='python3' alias pip='pip3'

Note that the above included an alias for pip to pip3 as well.

## 6.2 Create a virtual environment to use for Dev

A Python virtual environment is a development sandbox which allows for segmentation of development environments. This allows for management of different combinations and versions of Python releases, deployed packages, development and testing environments, and shifting between entirely different development projects. See the [Python documentation for venv](https://docs.python.org/3/library/venv.html) for a more detailed explanation and rationalization of Python virtual environments.

The use of Python virtual environments can't be overstated as a best practice to organize Python versions and package management. [PEP-405](https://www.python.org/dev/peps/pep-0405/)

Creating virtual environments to use in Python programming can be done entirely with packages that are include in the Python install or by additional third party applications.

The choice of tools to create and manage Python virtual environments is dependent on situation, preference, and use case.

### Tool/App Use Case

[venv](https://docs.python.org/3/library/venv.html) Implementation of the virtualenv as a Python module included in the Python install since v.3.3.

[virtualenv](https://virtualenv.pypa.io/en/latest/index.html) Includes features not included in venv (see the [comparison)](https://virtualenv.pypa.io/en/latest/index.html).

[Anaconda](https://www.anaconda.com/) A heavyweight package and virtual environment manager. It acts as an "all in one" for Python

application versions, package management, virtual environments, additional programming languages such as R and Julia and tools such as visualizers and IDEs. The full fledged install can overtax some systems and performance can suffer an overly ambitious installation configuration. A personal license for individuals is free for non-commercial use. Use by for-profit or governmental organizations with more than 200 people requires licensing.



[miniconda](https://docs.conda.io/en/latest/miniconda.html) A slimmed down version of Anaconda focused on virtual environment and package management, includes only conda and Python, not open source, but free.

[conda](https://docs.conda.io/en/latest/) The open source package manager utilized by Anaconda and miniconda.

[pip](https://pip.pypa.io/en/stable/) Included in Python 3.4 and later, this tool does not manage the virtual environment but does handle package management for both venv and virtualenv. Not all python tools have been integrated into the Anaconda repositories or packages for install. Some, such as mkdocs, must still be installed via pip even when using Anaconda3 or miniconda.

### 'virtualenv venv' vs. 'venv virtualenv'

Avoid the confusion of the typical example given in the documentation of **virtualenv** which uses the command executed as "virtualenv venv." This calls **virtualenv** to create a virtual environment named **venv**.

Compare with "venv virtualenv" which calls **venv** to create a virtual environment called **virtualenv**. For most practical purposes when using Python 3.6 or greater it doesn't matter whether **venv** or **virtualenv** is used to create the virtual instance.

But don't use "virtualenv venv." That's just obnoxious.

Other tools such as

[r](https://virtualenvwrapper.readthedocs.io/en/latest/)

virtualenvwrapp

[e](https://virtualenvwrapper.readthedocs.io/en/latest/)

,

pipe

[n](https://pipenv.pypa.io/en/latest/)

[v](https://pipenv.pypa.io/en/latest/)

,

[p](https://github.com/berdario/pew)

[e](https://github.com/berdario/pew)

[w](https://github.com/berdario/pew)

,

[t](https://tox.wiki/en/latest/)

[o](https://tox.wiki/en/latest/)

[x](https://tox.wiki/en/latest/)

and

[n](https://nox.thea.codes/en/stable/)

[o](https://nox.thea.codes/en/stable/)

[x](https://nox.thea.codes/en/stable/)

may be useful but are not currently utilized in this

project.

Pye

[n](https://github.com/pyenv)

[v](https://github.com/pyenv)

was deprecated in Python 3.5 and not utilized.

**Other Python Tools**



### 6.2.0.1 No Representations or Warranties

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 7 Project Package Requirements

7.1 Currently implemented packages:

This project utilizes the following packages, all of which should be available under an Open Source license via [Py](https://pypi.org/). List updated as of:

**02/16/2022 (February 16th, 2022)**

Make sure to source the virtual environment first and then '*pip | conda install* ' to bring in the package.

alabaster

argcomplete

argh

astroid

attrs

Babel

brotlipy

certifi

cffi

charset-normalizer

click

colorama

conda

conda-package-handling

cryptography

dnspython

docutils

elementpath

Faker

flake8

future

ghp-import

idna

imagesize

importlib-metadata

iniconfig

isort

Jinja2

joblib

lazy-object-proxy

livereload

lunr

Markdown

MarkupSafe

mccabe

mergedeep

mkautodoc

mkdocs

mkdocs-autorefs

mkdocs-bootstrap

mkdocs-material

mkdocs-material-extensions

mkdocs-print-site-plugin

mkdocstrings

mypy

mypy-extensions

nltk

packaging

pipx

platformdirs

pluggy

psutil

py

pycodestyle

pycosat

pycparser

pydocstyle

pyflakes

Pygments

pylint

pymdown-extensions

pyOpenSSL

pyparsing

PySocks

pytest

python-dateutil

python-dotenv

pytkdocs

pytz

PyYAML

pyyaml\_env\_tag

regex

requests

ruamel-yaml-conda

six

snowballstemmer

text-unidecode

toml

tomli

tornado

tqdm

types-requests

types-urllib3

typing\_extensions

urllib3

userpath

watchdog

wrapt

xmlschema

zipp

## **7.1.0.1 No Representations or Warranties**

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

8 Getting the code.

## 8.1 Pulling the code from GitHub

The e-Invoice Onboarding Toolkit is a public repository. The code is freely available under an MIT Licene for individuals and organizations to pull, review, and modify as they chose in order to further their participation with the project.

It is entirely possible to pull the code from github anonymously using a link from within the repo.

!

"

Look for the green "Code" button which will provide links to clone the code using https, ssh, the git desktop, or a zip

file.

!

"

This going to pull the repo at the root ./e-invoice-Onboarding-Toolkit.

!

"

Open the folder as a project within your IDE or editor of choice.

**Pulling the code from GitHub**



Those who would like to become more involved and want to do more than anonymously pull code can contribute by:

1. Creating a GitHub account if one hasn't already been created.
2. Installing the GitHub CLI or the GitHub desktop application
3. Configuring the personal profile and Secure Shell (SSH) keys to securely submit code to the repository.

### 8.1.0.1 No Representations or Warranties

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

I. Using the Code

# 9 Integrating Code Modules

## 9.1 The Package Structure and Using the Modules

How the package is organized.

The top level of the packages is named "einvoice." It is the parent to all other packages and modules.

**Directory Structure**

**dot Notation**

./einvoice

einvoice

There are two sub-packages called discovery and delivery. A directory named "test" contains unit tests for both delivery and discovery. A third directory named "docs" is also at this level and contains project documentation.

**Directory Structure**

**dot Notation**

./einvoice

einvoice

./einvoice/discovery

einvoice.discovery

./einvoice/delivery

einvoice.delivery

./einvoice/test

einvoice.test

./einvoice/docs

NA - does not contain code artifacts

Third level directory contains the Python modules containing actual application code. Within the discovery directory there is a "conf" directory intended for application configuration work, a "data" directory for files and applications to generate test data and scenarios.

**Directory Structure**

**dot Notation**

./einvoice

einvoice

./einvoice/discovery

einvoice.discovery

./einvoice/delivery

einvoice.delivery

./einvoice/test

einvoice.test

./einvoice/docs

NA - does not contain code artifacts

**Directory Structure**

**dot Notation**

./einvoice/discovery/conf

einvoice.discovery.conf

./einvoice/discovery/data

einvoice.discovery.data

./einvoice/discovery/accessor.py, app\_handler.py,

app\_logging.py, create\_tracking\_id.py, dns\_query.py,

einvoice\_message\_package.py, line\_item.py,

party\_address.py, semantic\_model.py, smp\_query.py,

urn\_hasher.py, urn.py

einvoice.discovery.accessor.Accessor,

einvoice.discovery.app\_logging.create\_logger, etc.

./einvoice/delivery/import\_xsd

einvoice.delivery.import\_xsd.ImportXSD

./einvoice/delivery/tests

einvoice.delivery.tests

Fourth level directories are the deepest in the application and contain code in the conf and data directories.

**Directory Structure**

**dot Notation**

./einvoice

einvoice

./einvoice/discovery

einvoice.discovery

./einvoice/delivery

einvoice.delivery

./einvoice/test

einvoice.test

./einvoice/docs

NA - does not contain code artifacts

./einvoice/discovery/conf

einvoice.discovery.conf

./einvoice/discovery/data

einvoice.discovery.data

./einvoice/discovery/accessor.py,

app\_handler.py, app\_logging.py,

create\_tracking\_id.py, dns\_query.py,

einvoice\_message\_package.py, line\_item.py,

party\_address.py, semantic\_model.py,

smp\_query.py, urn\_hasher.py, urn.py

einvoice.discovery.accessor.Accessor,

einvoice.discovery.app\_logging.create\_logger, etc.

./einvoice/delivery/import\_xsd.py

einvoice.delivery.import\_xsd.ImportXSD

./einvoice/discovery/conf/config\_tool.py,

smp\_config.py

einvoice.discovery.conf.config\_tool.EInvoiceConfig, etc.

**Directory Structure**

**dot Notation**

./einvoice/discovery

/data/create\_sample\_data.py

einvoice.discovery.data.create\_sample\_data.CreateSampleData

./einvoice/discovery/tests

/test\_app\_logging.py

einvoice.discovery.tests.test\_app\_logging.test\_log\_creation

./einvoice/discovery/tests

/test\_app\_logging.py

einvoice.delivery.tests.test\_app\_logging.test\_import\_xsd

## 9.2 Additional Files

Additional files included in the project which are important.

**File**

**Purpose**

./einvoice/delivery/app.log

Application log created by app\_logging.py for delivery sub-package.

./einvoice/delivery/web\_response.log

Response logging to feed into a webservice for delivery sub-package.

./einvoice/discovery/app.log

Application log created by app\_logging.py for discovery sub-package.

./einvoice/discovery

/web\_response.log

Response logging to feed into a webservice for discovery sub-package.

./einvoice/docs

Markdown files compiled into the project documentation.

./einvoice/docs/jupyterlab

Stored JupyterLab sandboxes which may be shared via Google Colab or

downloaded and run on a Jupyter service instance.

./einvoice/docs/pdf

Stored PDF files (entity diagrams) which may be included in the

documentation.

./einvoice/docs/drawio

Stored PDF files (vector graphic diagrams) which may be included in the

documentation.

./einvoice/discovery/tests/ \*.sh

An assortment of shell scripts to run various linters on the modules.

Includes pylint, mypy, flake8, pycodestyle, pydocstyle, and combinations.

./einvoice/discovery

/data/item\_list.csv, per\_item\_list.csv

CSV files which contain same data values to populate an einvoice.

.einvoice/.env .env.example.dev

Configuration files which contain example values for testing purposes.

ebms-header-3\_20220119.xsd,

sample\_msg.xml

XSD containing schema definition for ebMS header and a sample

message to test against.

9.3 Note on classes with modules.

**All module code is in classes and methods.**

All code in the discovery and delivery sub-packages is encapsulated in a class and a method within a class. There are no executable functions outside of a class.



*There is no entry point to execute this code and instantiate any of the classes or methods at the command line.*

Examples of implementing and executing the code can be found in the [test cases](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#test_cases), [start-to-finish example](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#start_to_finish), or the [JupyterLab/Notebooks](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#google_colab_pages).

Test cases are not encapsulated in classes or methods but are instead named functions.

The code is as Pythonic as possible in naming files for exactly what they do. The functionality can be broken down as:

!"Dataclasses - modules which define some of the key entities at use in the project.

!"urn.py - Dataclass for a URN.

!"semantic\_model.py - Dataclass for the semantic model (the einvoice itself).

!"party\_addresss.py - Dataclass for a party entity within the Four-Corner model.

!"line\_item.py - Dataclass for a line item on the semantic model (einvoice).

!"einvoice\_message\_package.py - Dataclass to contain all the information to be transmitted, i.e., the payload, in the einvoice message.

!"Specific workflow actions - modules which execute specific tasks within the process workflow.

!"urn\_hasher.py - takes the inputs of the party\_id, specification, and schema\_id and creates the NAPTR look-up uri.

!"dns\_query.py - take the NAPTR look-up uri and execute it against DNS. The output is the SMP uri and the existing urn is passed forward as well.

!"smp\_query.py - receives the SMP uri and urn and creates two REST API calls to the endpoint based on the inputs. Executes the webservice calls and receives a response. Parses the response and returns it as a string containing the URI of corner 3 in the model.

!"import\_xsd.py - takes as an input an XML file and checks its validity against an XSD. In this case it is the XML of an ebMS message header checked against an AS4 conformance profile.

!"Other "helper" modules -

!"accessory.py - module to run the "Start-To-Finish" process, execute via test scripts.

!"create\_sample\_data.py - construct sample data entities to use in testing the semantic model.

!"create\_tracking\_id.py - create an arbitrary id with a given configuration to use to track the message through the process. Could be used in lieu of a UUID.

!"app\_handler.py - module closest to being an executable form the command line. A prototype module to run the "Start-to-Finish" directly, if all required configuration is complete.

!"app\_logger.py - a custom logging implementation to be used by all the other modules, including test modules, to standardize output and aggregate to single stream each for application logging, to system out, and response to a webservice.

### 9.3.0.1 No Representations or Warranties

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 10 e-Invoice Onboarding Tool-kit

## 10.1 Working with the code

### 10.1.1 Test Cases

Every module includes a test module in the einvoice/test directory.

The test cases are written as functions and

**can**

be directly called from the command line.

The use case for the test cases include using Pytest.

**The Pytest package must be installed in your Python distribution.**

**Test cases are written as functionsversus classes.**



**Use Pytest to run the test cases.**



From the a terminal console, change directory into the ./einvoice/test directory.

To see the list of available test files, either in your file browser or via the command line ("dir .\einvoice\tests\" or "ls -al ./einvoice/tests/").

**The ./test directory must be at the same level as the code.**

Out of the box, Pytest requires that without additional configuration it must be executed from a directory at the same level of the code that's being tested. That is, test scripts are in ./einvoice/test and code files are in ./einvoice/discovery and ./einvoice/delivery.



Pytest will automatically look for files formatted as test files, with "test" in the lead of the filename. To execute an individual test the syntax is:

pytest test\_app\_logging.py

No test is dependent on any other, and each may be run on its own, or run them all at once, in any order.

The test will run and either the "assert" statement(s) inside will pass or fail.

Failures MUST be resolved prior to attempting to check code into GitHub as our baseline CI/CD process checks for these failures before committing and will not continue if any are found.

The included assert statements currently test a variety of cases up to validation of urn creation, query of the NAPTR DNS record, REST API call to the SMP, and validation of the ebMS header against the AS4 conformance profile.

The test folder also contains a number of shell scripts to validate the code using a number of linters including flake8, pylint, mypy, pycodestyte, and pydocstyle. Prior to check in, all code has had warnings from all linters resolved or noted.



*Figure*

*1*

*:*

*Successful completion of all test cases.*

#### 10.1.1.1 No Representations or Warranties

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 11 Start-to-Finish

## 11.1 Start-to-Finish Integration

There is not currently an "end-to-end" test for the e-Invoice Four-Corner Model to validate the workflow in its entirety.

The next best use case is a "Start-to-Finish" of the discovery process.

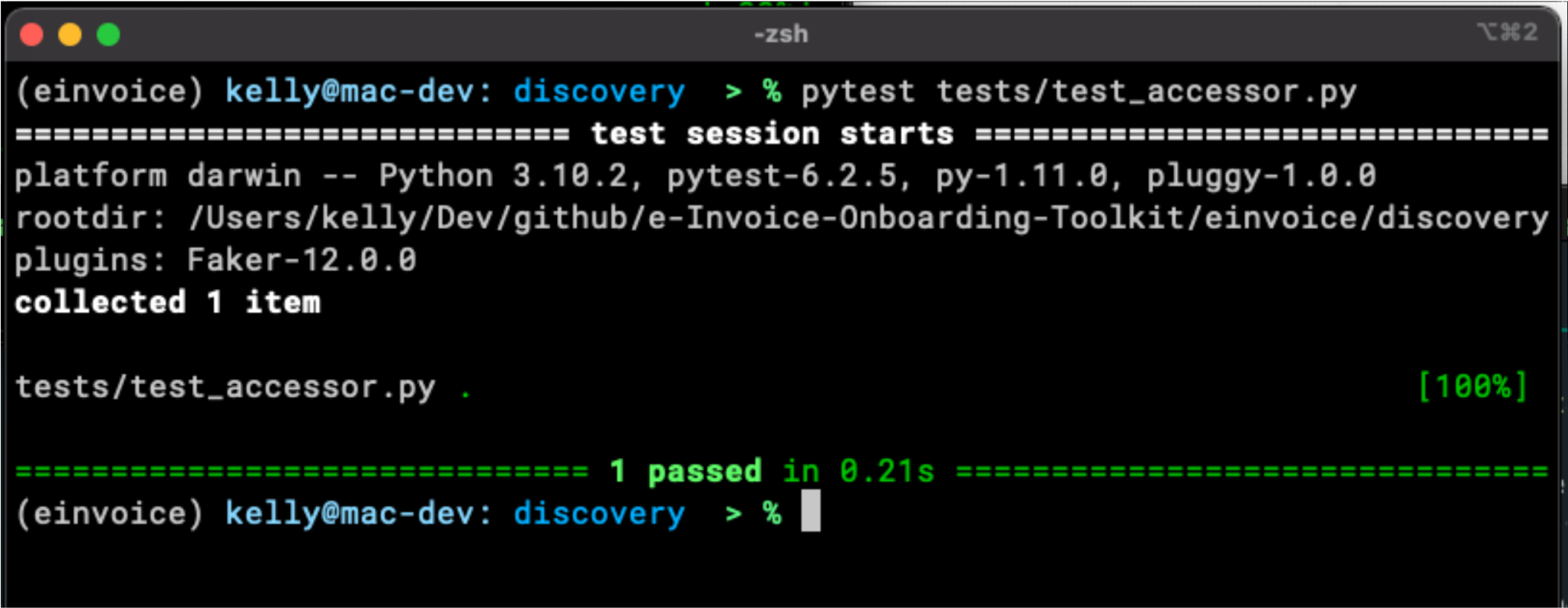
This "Start-to-Finish" entails validating:

1. The hashing functionality to derive the URNN for look-up from the specification, the party ID, and the schema ID.
2. Executing the DNS NATPR lookup and extracting the relevant SMP URI.
3. Constructing the two REST requests including the **smp service group url** and the **smp service url**.
4. Executing the two REST requests to the SMP server.
5. Extracting the Corner 3 endpoint URI from the response from the SMP server.

This functionality is provided in the accessor.py module and validation is done in a single test case called test\_accessor.py.

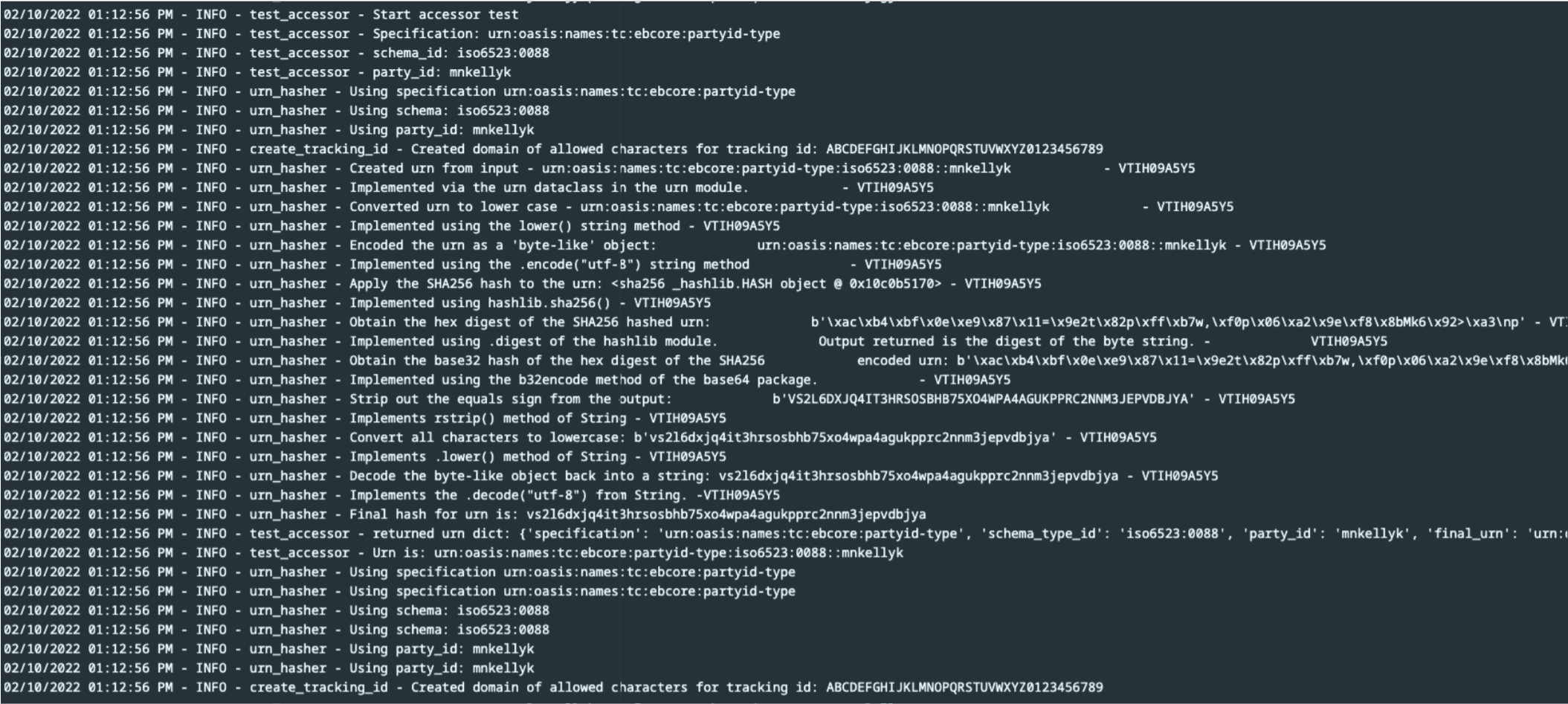
Execute the "Start-to-Finish" test as reference in the [more detailed instructions for running the test cases](https://bpc-opensourcetools.github.io/e-Invoice-Onboarding-Toolkit/print_page/#test_cases).

./einvoice/discovery/pytest tests/test\_accessor.py



*Figure 2: Successful completion of "Start-to-Finish" test ca*

Further review and analysis of the Start-toFinish process can be found in the app.log which for the accessor.py module resides in the ./einvoice/discovery directory.



*Figure 3: Successful completion of "Start-to-Finish" test case recorded in the app.log file.*

### 11.1.0.1 No Representations or Warranties

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 12 Jupyter Notebooks on Google Colab

## 12.1 Colab Sandboxes

[JupyterLab](https://jupyter.org/) is a sandbox development environment which allows for, among other things, rapid prototyping or testing of small units of code. They provide a framework to execute code without building a whole application or even a complete module.

Most of the code already incorporated into the project started out in a JupyterLab runtime environment.

JupyterLab is also great for documenting and demonstrating exactly what's going on with a piece of code.

JupyterLab artifacts worked on for the project are stored as static documents in GitHub in the e-InvoiceOnboarding-Toolkit project under ./einvoice/docs/jupyterlab.

[Google Colab](https://colab.research.google.com/) pages implement [JupyterLab](https://jupyter.org/) runtime with live sandbox environments. Pages can be linked from the e-Invoice-Onboarding-Toolkit GitHub repository, or pulled from the repository and saved locally by anyone with a Google account.

12.1.1 urn hashing and DNS NAPTR lookup.

Colab page with examples of how to hash the specification, the schema\_id, and the party\_id to create the urn and perform the natpr dns query is at this [Colab page](https://colab.research.google.com/drive/1kfMedMUapeaOS6u9hnS8IcmQaury8znP?usp=sharing). Examples 6, 7, 8, and 9 run the hash and submit against DNS in real-time.

The JupyterLab file is: **urn\_hash\_work.ipynb**.

## 12.2 SMP query

Colab page with examples of how to transform the urn and party\_id and submit it to the SMP uri is at this [Colab page](https://colab.research.google.com/drive/14EVSc0GyjU0H9776UEXqoEoD3x5Kn2RV?usp=sharing).

They JupyterLab file is: **smp\_url\_transformations.ipynb**.

## 12.3 ebMS Message Header validation

Colab pages with examples of reading an XSD file and validating an XML file has two Google Colab pages for different aspects of the work.

Inspection and validation of the XSD file has this [Google Colab Page](https://colab.research.google.com/drive/1zuPcP1ofEe8PReew9KbGY13EQJsNN8Es?usp=sharing). The JupyterLab file is: **ebMS XML 3 schema.ipynb**.

Validation of an xml file against the XSD is done using this [Google Colab Page](https://colab.research.google.com/drive/1ExMZUD_5larW0wEuGJ5erHkFuoozcLCF?usp=sharing) The JupyterFile is: **Validate\_bdxas4.ipynb**.

For ease of access these files are copies stored on the drive of one of the project Developers and is free and open to anyone to view and run. Interested individuals should make copies of the Labs for themselves and run on Google Colab under their own account or an instance of JupytyerLab running on Anaconda, VS Code, or a Python install.

### 12.3.0.1 No Representations or Warranties

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 13 Additional Infrastructure Build-out

The code is intended to interact with other participants in the Four-Corner Model, including Access Providers, DNS servers and SMP service providers.

## 13.1 SML

**Q:**

How do you test the toolkit? How do you create a NAPTR DNS record entry on a domain?

**A:**

In order to do theperform an SML look-up, the appropriate NAPTR records must be in place. The assumption is that a NAPTR DNS record exists as a key:value pair. A look-up of the "key" in the DNS of the NAPTR record will return the "value." The "key" is the hashed value of the urn. The value being sought and returned is the URI of the SMP for the next step in the Model.

Access Point 1 in Corner 2 may be acting in the role of the SML and handling tasks associated with it. These tasks could include: \* Creating the URN \* Creaing the hash value of the URN \* Queryhing the DNS NATPR record record \* Returning the SMP URI

The Python modules provide examples of some ways the tasks of SML could be accomplished, either as, or by, an Access Point or an organization on its own behalf.

For testing purposes the BPC has is running a DNS server on Amazon Web Services Route53 on the test domain of sc-b2b.us. This allows for the registration of urn hashes in the DNS domain of sc-b2b.us. These entries are live in the DNS accessible worldwide.

A REST API is available at <https://sml-api.sc-b2b.us/docs> to register SML entries, which are the DNS NATPR records on the sc-b2b.us domain. This process creates the urn hash based in inputs provided by the user. Organizations wishing to register a test urn on the BPC test domain can reach out to the BPC or @mnkellyk for assistance in using the web interface.

Once these SML/DNS NATPR entries are created, they can queried using the toolkit to make public queries to DNS NAPTR look-up within a matter of seconds.

The code implemented to create the NAPTR DNS record on AWS Route53 is available in the GitHub repository: [BPCOpenSourceTools/sml-service-r53](https://github.com/BPC-OpenSourceTools/sml-service-r53).

## 13.2 SMP

The BPC also has an application to test the SMP REST API service calls on the same domain as the SML at <https://smp-api.sc-b2b.us/docs> . Again, this is a REST API to make web service calls to test the toolkit.

The SMP itself s a web service queried by a SOAP API call to return the Corner 3 URI or terminal endpoint. The specification for the actual API can be found in the document: [Service Metadata Publishing (SMP) Version 2.0](http://docs.oasis-open.org/bdxr/bdx-smp/v2.0/bdx-smp-v2.0.html) dated 14 February 2021 as an Oasis standard.

**Section 5.4 Resources**

**Resource**

**URI**

**Method**

**XML resource root**

**element**

**HTTP**

**Status**

**Description of**

**returned**

**content**

test

test

test

test

test

testz

ServiceGroup

./bdxr-

smp-2/[{identifier

scheme}::]

}

{

participant id

See section 3.6 for

{

}

participant id

format

GET

<

ServiceGroup

>

200

; 500;

404

Holds the

Participant

Identifier of the

recipient, and a

list of references

to individual

ServiceMetadata

resources that

are associated

with that

participant

identifier.

ServiceMetadata

./bdxr-

smp-2/[{identifier

scheme}::]

{

participant

id}/services

/{service ID} See

section 3.7 for

{

service ID

}

format

GET

ServiceMetadata

<

>

200

; 500;

404

Holds all of the

metadata about

a Service, or a

redirection URL to

another Service

Metadata

Publisher holding

this information.

The SMP service provided by the BPC registers a urn for query. Note that the API specification is essentially the urn with modifications to include some additional service capability codes but mostly to accommodate characters that would otherwise be illegal in a URL.

After registering a urn(s) on the SML service, go to the SMP service at <https://sml-api.sc-b2b.us/docs> to register the urn there in order to get a response for testing SMP query functionality.

The code implemented to create the NAPTR DNS record on AWS Route53 is available in the GitHub repository: [BPCOpenSourceTools/smp-service](https://github.com/BPC-OpenSourceTools/smp-service).

**Client/server certificate trust implicit in the test domain.**

The SMP service specification has rules regarding Access Point and Requester security certificates (client) and SMP provider (server side) security certificates to establish non-reputability, etc. The BPC e-Invoice workgroup has not fully addressed the question of how the security certificate relationships are going to be managed. As such, the test application does not validate the client/requestor or its Access Point. It does return the security certificate of the server in its response, registered under letsencrypt.org. In the future, if rules were to be finalized, the security certificate relationship would need to be incorporated into the SMP REST API protocol.



### 13.2.0.1 No Representations or Warranties

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 14 Project Roadmap

## 14.0.1 Discovery #1 - SML NAPTR DNS Lookup

1. Feature – Access Point A sends UNAPTR DNS query with a Party ID/Party ID Schema hash and obtains a valid response with connection information to the SMP.

**Action**

**Actor**

**Scoped?**

Generate the request to Access Point A, which includes Party ID, Party

ID Schema, Invoice Data

Seller

No

Transform and format contents of Seller’s request to create UNAPTR

DNS query

Access Point A Yes

Query DNS

Access Point A

Yes

Return response to query

SML – Reply from UNAPTR

DNS query.

No

Receive query response from DNS, which is the URI to the SMP

Access Point A

Yes

## 14.0.2 Discovery #2 - SMP REST API Query

1. Feature –Access Point A sends a REST API query to the SMP URI to obtain a valid response with connection information of a target Access Point and invoice capabilities of the customer.

**Action**

**Actor**

**Scoped?**

Send response with SMP URI to Access Point.

SML

No (not in this

feature)

Create REST query to service provider to obtain buyer’s service

capabilities.

Access Point

A

Yes

Send REST query to service provider to obtain participant’s service

capabilities.

Access Point

A

Yes

Receive query and send response with Sellers Capabilities and route to

endpoint.

SMP

No

Receive response to query of participant’s capabilities.

Access Point

A

Yes

## 14.0.3 Delivery - AS4 Message Exchange

1. Feature – An invoice with a semantically correct format is delivered using AS4 protocol.

**Action**

**Actor**

**Scoped?**

Compose semantically correct e-Invoice based on response from service provider

about participant’s capabilities

Access Point

A

Yes

Format e-Invoice in compliant AS4 format

Access Point

A

Yes

Send e-Invoice to final destination obtained from SMP service provider

Access Point

A

Yes

Receives the request

Access Point

B

No

### 14.0.3.1 No Representations or Warranties

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

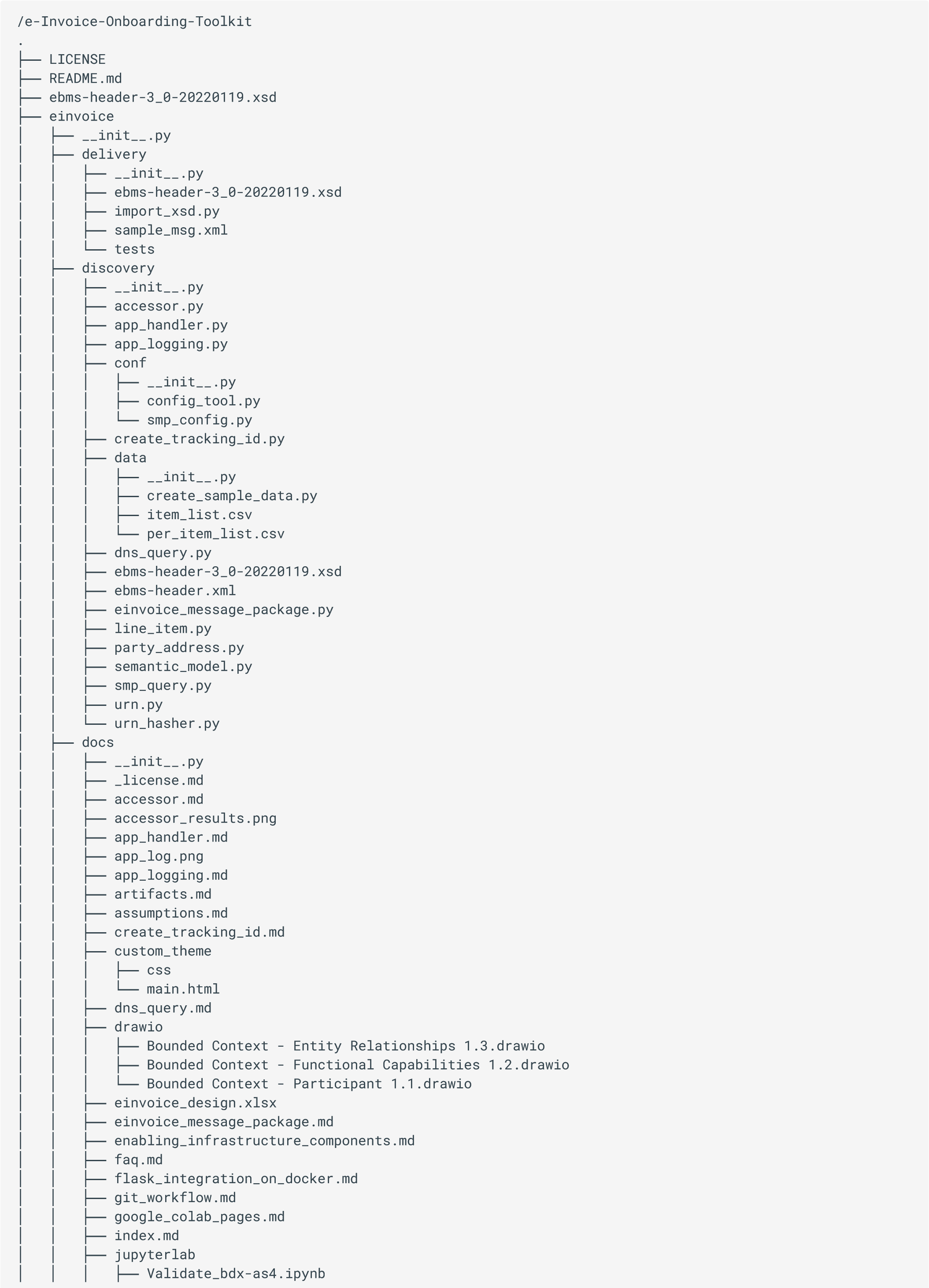
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 15 The Repo(sitory)

## 15.0.1 Repository Layout

This project includes the following files, *i.e., "artifacts"* in **.py**, .**md**, .**txt** and other formats, updated as of:

**02/16/2022 (February 16th, 2022)**





### 15.0.1.1 No Representations or Warranties

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 16 Sample git workflow

16.0.1 A minimal git "script" to work with the code

Below is a sample workflow of a very rudimentary process to create a branch in Github, add code, and push up to the repo on Github.

1. Create a new branch:

git checkout -b <insert branch name here`>

1. Implement your changes
2. Add into the repo:

git add . git commit -m <your comment here> git push

:pushes your changes up to the remote branch

1. Either create a pull request in Github, or:

git checkout main

git merge <branch you want to merge here> git push to push main changes up to remote branch

## **16.0.1.1 No Representations or Warranties**

This software is free and Open Source offered under an MIT license. The developers of the software make no representations or warranties as to the software or its fitness for a particular purpose. This code is meant for educational and research purposes only. The code is offered "as-is" and is not intended to be used in a production environment. It is intended for developers of software related to the 4-corners Model to use as a stepping-off point for further development efforts.

# 17 OASIS Resources

Reference Links:

## 17.1 Documents

**OASIS Specifications**

OASIS ebXML Messaging Services Version 3.0: Part1, Core Featur

[e](http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/os/ebms_core-3.0-spec-os.html)

[s](http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/os/ebms_core-3.0-spec-os.html)

OASIS Standard, October 1, 2007, has:

OASIS ebXML Messaging Services Version 3.0: Part 2, Advanced Featur

[e](https://docs.oasis-open.org/ebxml-msg/ebms/v3.0/part2/201004/ebms-v3-part2-cd-01.html)

[s](https://docs.oasis-open.org/ebxml-msg/ebms/v3.0/part2/201004/ebms-v3-part2-cd-01.html)

dated June 30, 2010 with Normative Reference

of:

ebXML Messaging Services Version 3.0: Part 2, Advanced Featur

[e](https://docs.oasis-open.org/ebxml-msg/ebms/v3.0/part2/201004/rddl-ebms3-part2.html)

[s](https://docs.oasis-open.org/ebxml-msg/ebms/v3.0/part2/201004/rddl-ebms3-part2.html)

which is referenced by the following three

namespace URIs.

OASIS ebXML Messaging Services 3.0 Conformance Profil

[e](https://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/20707/ebms3-confprofiles-cs-01.html)

[s](https://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/20707/ebms3-confprofiles-cs-01.html)

, Committee Specification 1, dated April 24, 2010 references

the same namespace URI of http://docs.oasis-open.org/ebxml-msg/ns/ebms/v3.0/profiles/200707.

AS4 Profile of ebMS 3.0 Version 1

[.](http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/os/AS4-profile-v1.0-os.html)

[0](http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/os/AS4-profile-v1.0-os.html)

dated January 23, 2013.

Service Metadata Publishing (SMP) Version 2

[.](http://docs.oasis-open.org/bdxr/bdx-smp/v2.0/bdx-smp-v2.0.html)

[0](http://docs.oasis-open.org/bdxr/bdx-smp/v2.0/bdx-smp-v2.0.html)

dated 14 February 2021 as an Oasis standard.

## 17.2 XML Schema Definitions

**XSD Files**

**XSD File**

:

ebms-header-3\_0-2007

[0](https://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/os/ebms-header-3_0-200704.xsd)

[4](https://docs.oasis-open.org/ebxml-msg/ebms/v3.0/core/os/ebms-header-3_0-200704.xsd)

including snippets of sample XML and full SOAP for message headers. Note that

the Namespace URI identified in Part1 is incorrect and returns an error message.

**XSD File**

:

XSD for Routing Input reference paramet

[e](http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/part2/201004/ebms-multihop-1_0-200902_refactored.xsd)

[r](http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/part2/201004/ebms-multihop-1_0-200902_refactored.xsd)

**XSD File**

:

MessageFragment X

[S](http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/part2/201004/mf.xsd)

[D](http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/part2/201004/mf.xsd)

**XSD File**

:

Refactored Core Messaging X

[S](http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/part2/201004/ebms-header-3_0-200704_refactored.xsd)

[D](http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/part2/201004/ebms-header-3_0-200704_refactored.xsd)

## 17.3 Namespace URIs

**Namespaces**

*Namespace URI*

:

ebXML Messaging Services Version 3.0 Core Feature

[s](https://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/)

[)](https://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/)

and references:

*Namespace URI*

:

ebXML Messaging V3 Part 2: Multihop Routi

[n](http://docs.oasis-open.org/ebxml-msg/ns/ebms/v3.0/multihop/200902)

[g](http://docs.oasis-open.org/ebxml-msg/ns/ebms/v3.0/multihop/200902)

### Namespaces

*Namespace URI*: [ebXML Messaging V3 Part 2: Message Fragments](http://docs.oasis-open.org/ebxml-msg/ns/v3.0/mf/2010/04/)

*Namespace URI*: [Resource Directory Description Language(RDDL) 2.0](http://www.openhealth.org/RDDL/20040118/rddl-20040118.html) and references the XSD files below.

### TBD

Interoperable components under the AS4 Usage Agreements as specified in Section 5.2 of the [AS4 Profile of ebMS](http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/profiles/AS4-profile/v1.0/os/AS4-profile-v1.0-os.html) are non-normative. (Excepting that the new AS4 Interoperability Profile for Four-Corner Model networks does attempt to make interoperation normative via P-Mode configuration.) Samples of SOAP including XML, XMLNS and XSLT are included in Appendix A of the AS4 Profile.

It can be inferred from the ebXML Messaging V3 (ebMS3) namespace at [http://docs.oasis-open.org/ebxml-msg/ns /ebms/v3.0/profiles/200707](http://docs.oasis-open.org/ebxml-msg/ns/ebms/v3.0/profiles/200707) that a namespace URI for the November 12th, 2021 Committee Conformance Profile specification would be called something similar to https://docs.oasis-open.org/bdxr/ns/bdx-as4/v1.0/profiles/202112. That namespace or similar guidance has not yet been articulated by the Committee.



### 17.3.0.1 No Representations or Warranties

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE

WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# 18 MIT License

Copyright (c) 2022 BPC

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED

TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT

SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.