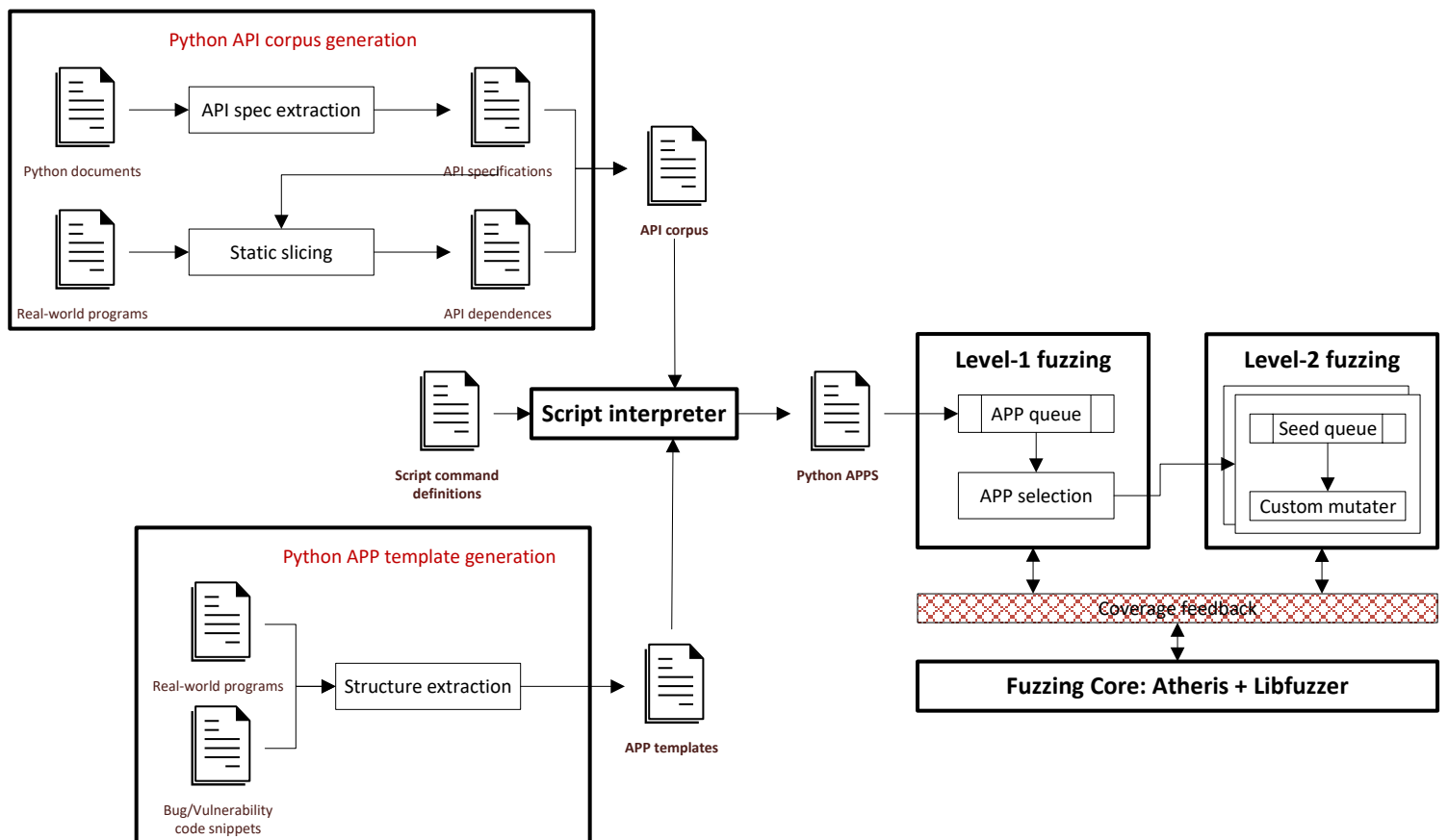
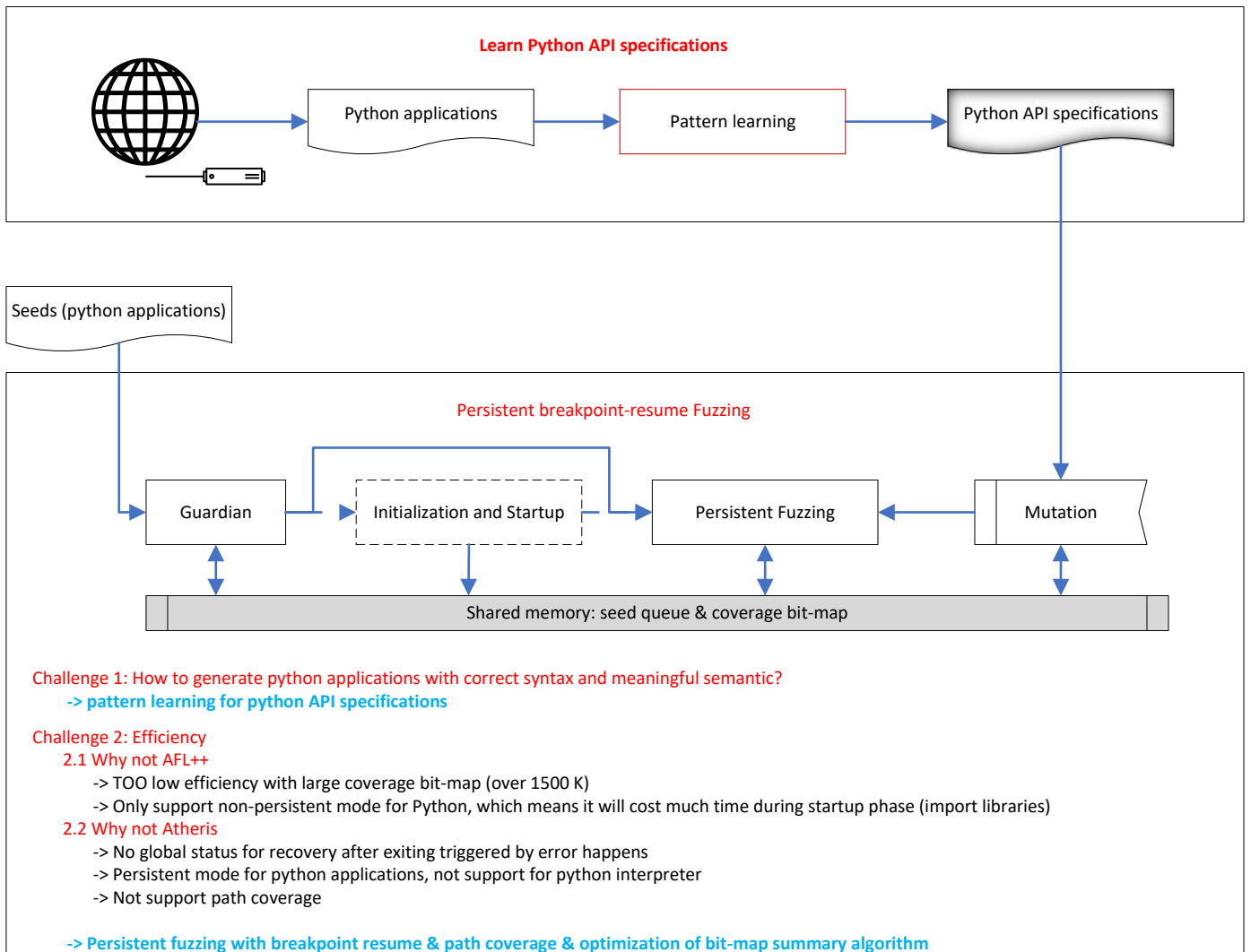
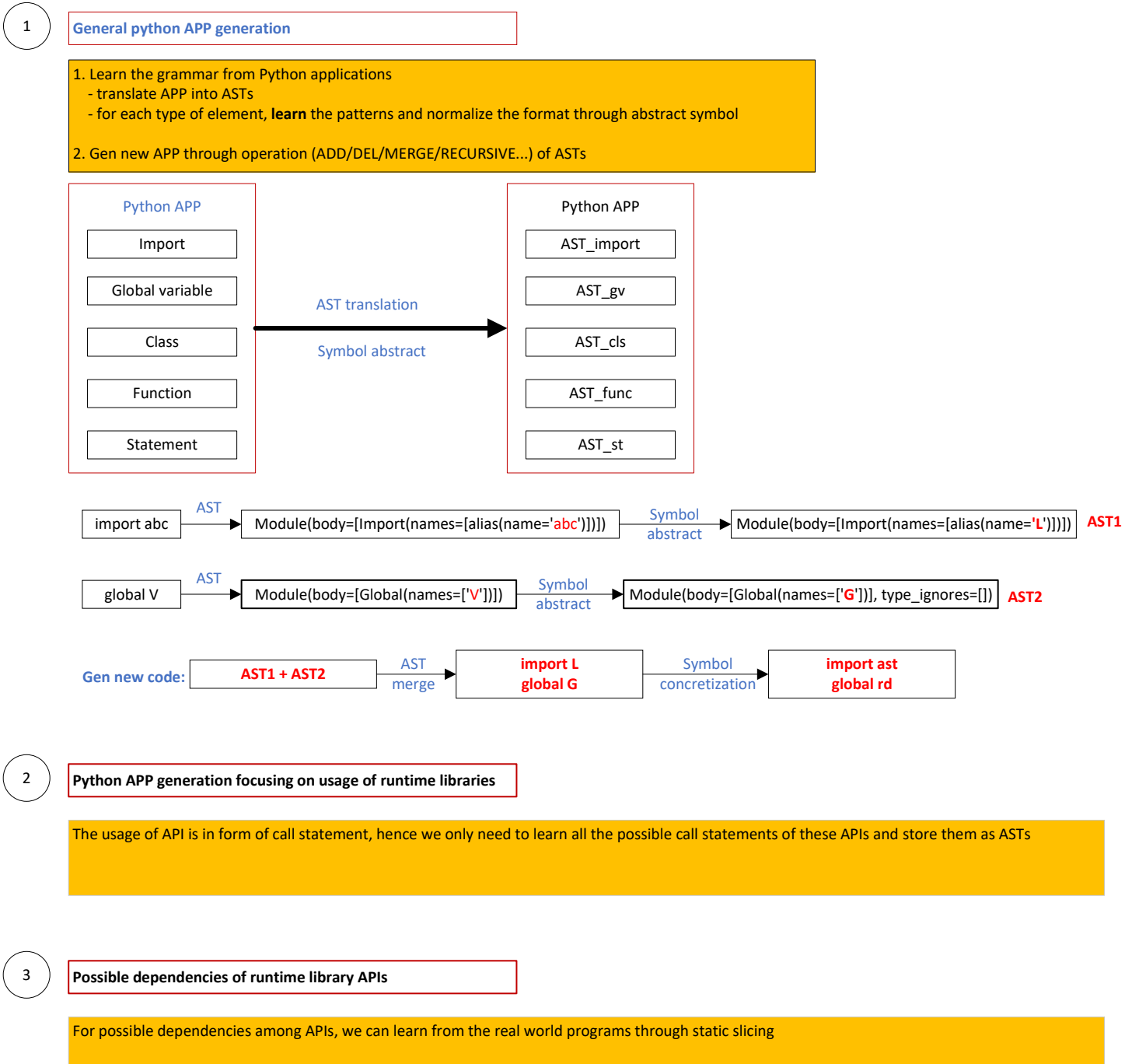


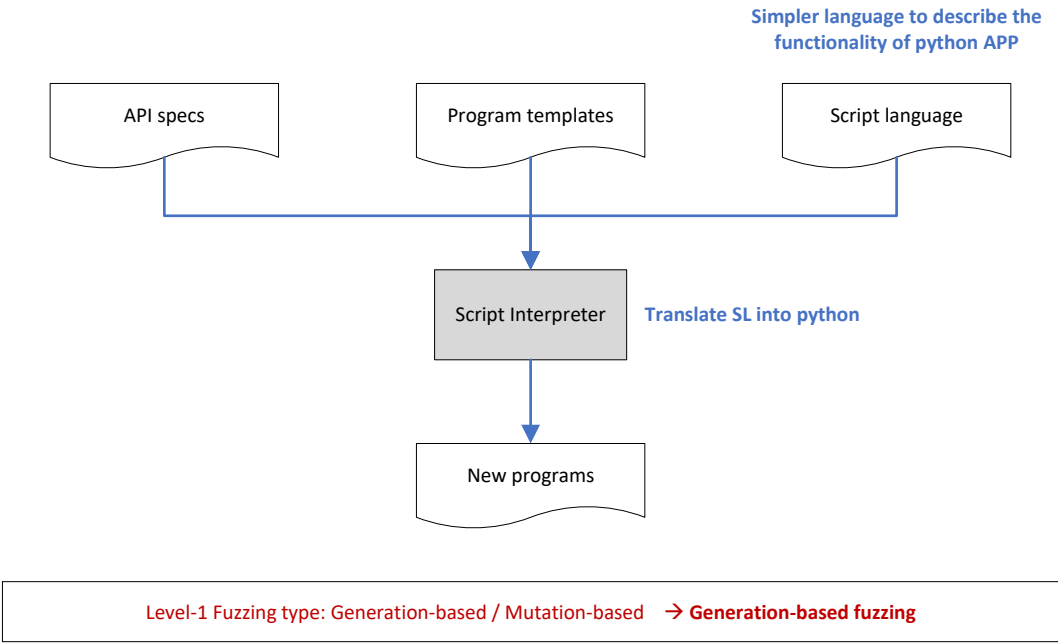
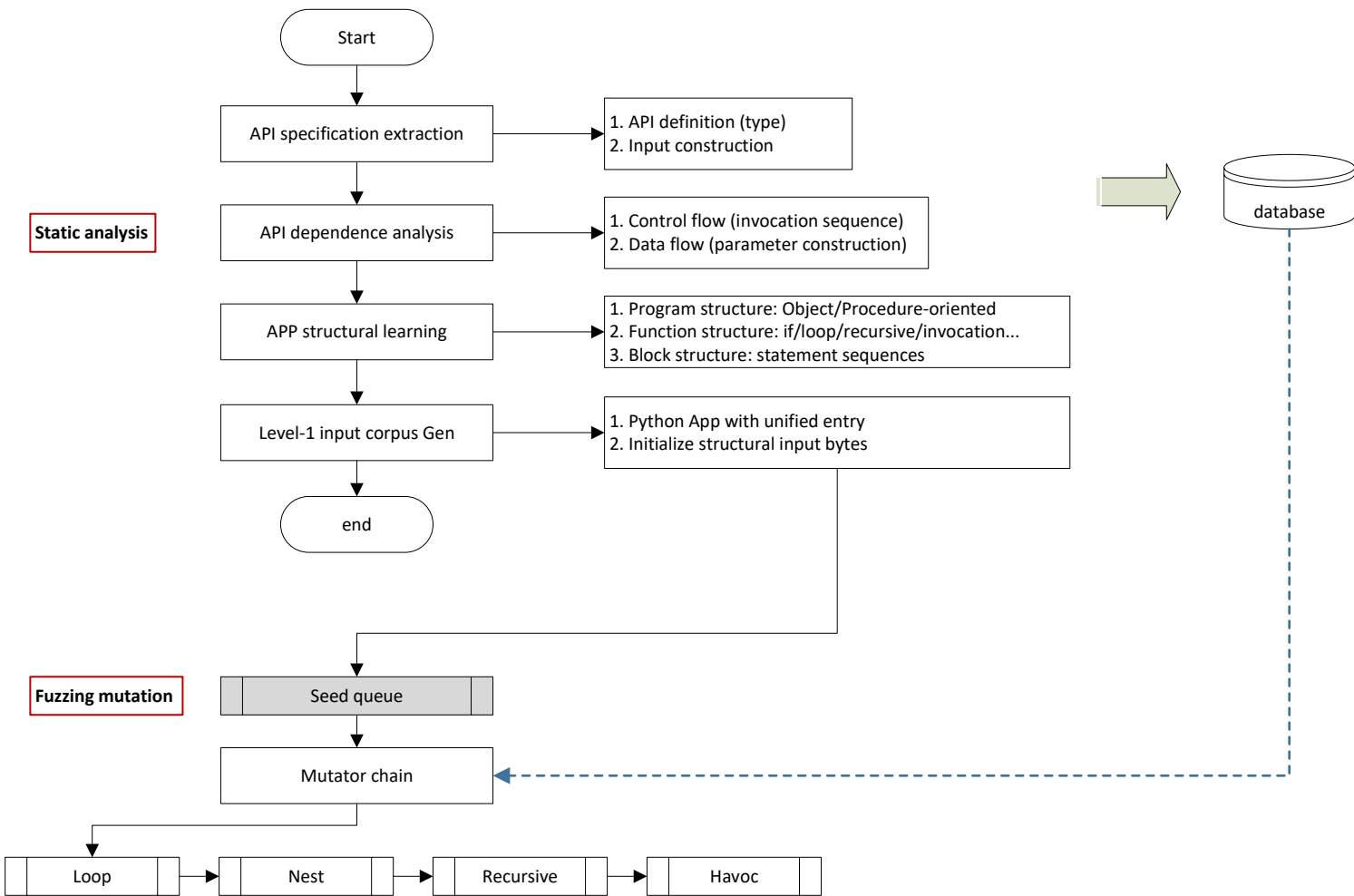
Overview



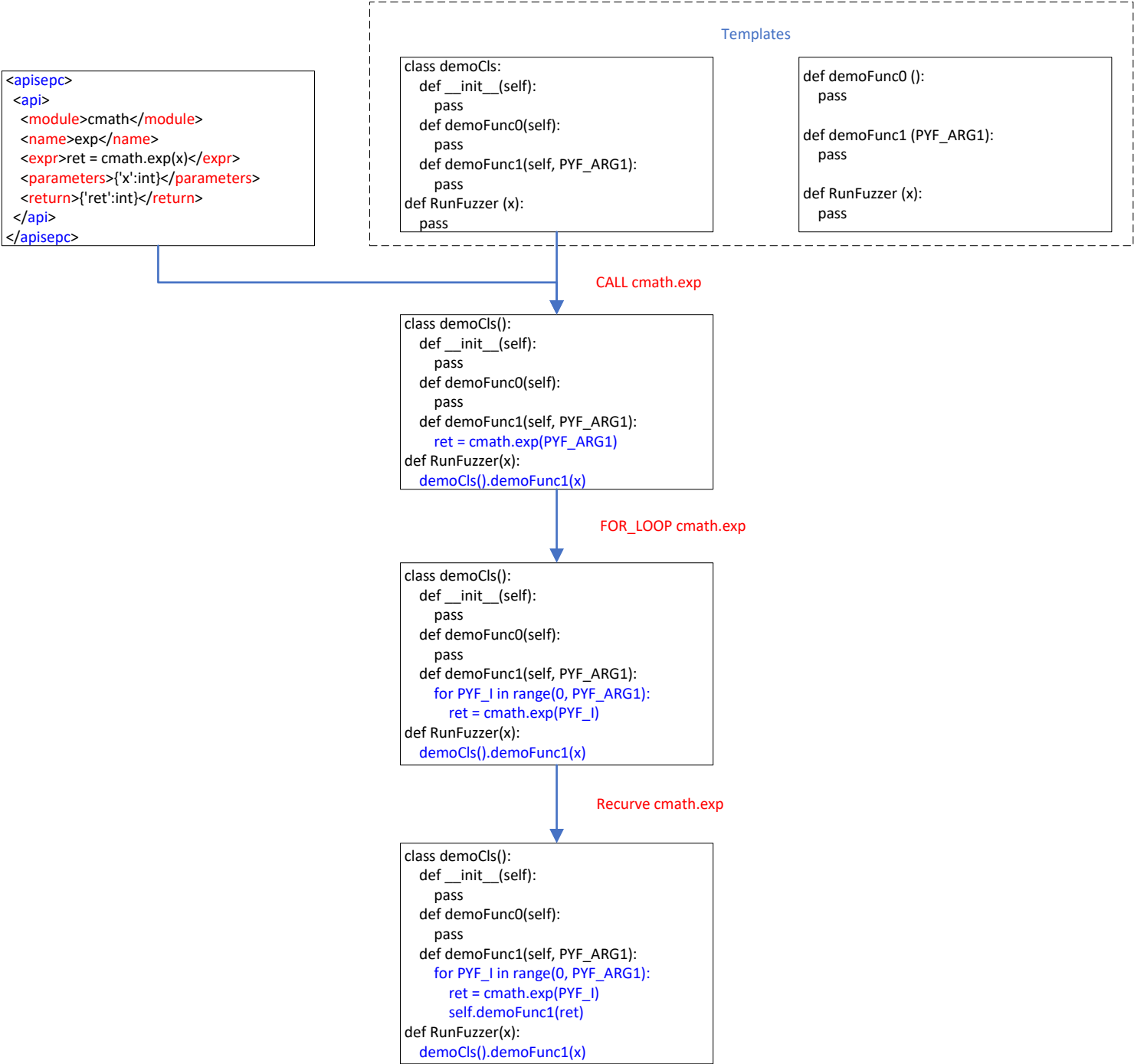
Python APP generation



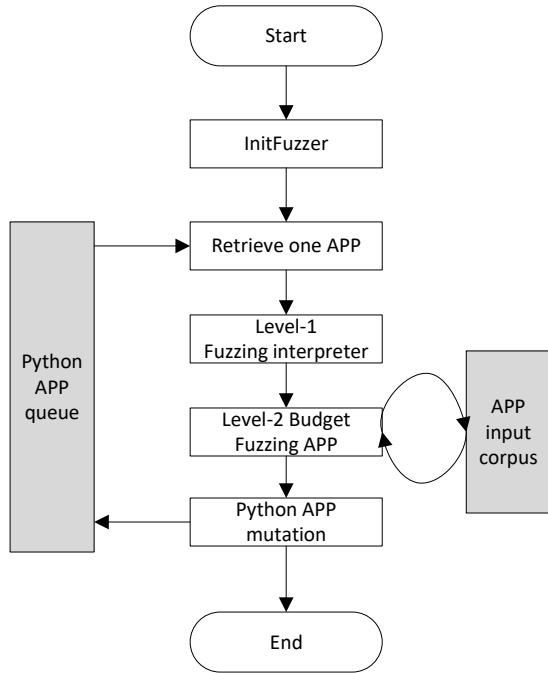
Software design



Example



Two-level Fuzzing



Level-1 Fuzzing: Targeting interpreter Core. We use infinite budget at level-1.

Level-2 Fuzzing: Targeting runtime libraries. We use finite budget at level-2 until no favored path/block/feature found.

```
<apisepc>
  <library name="Email">
    <module name="charset">
      <class name="Charset">
        <api>
          <name>get_body_encoding</name>
          <expr>ret = get_body_encoding()</expr>
          <parameters>{}</parameters>
          <return>{'ret':base64|7bit}</return>
          <dependencies> </dependencies>
        </api>

        <api>
          <name>header_encode</name>
          <expr>header_encode(str)</expr>
          <parameters>{'str':string}</parameters>
          <return>{}</return>
          <dependencies> </dependencies>
        </api>
      </class>
    </module>

    <module name="contentmanager">
      <api>
        <name>header_encode</name>
        <expr>header_encode(str)</expr>
        <parameters>{'str':string}</parameters>
        <return>{}</return>
        <dependencies> </dependencies>
      </api>
    </module >

    <errors>
      <exception>MessageError</exception>
      <exception>MessageParseError</exception>
      <exception>BoundaryError</exception>
      <exception>MultipartConversionError</exception>
    </errors>
  </library>
</apisepc>
```

Object-oriented

```
class demoCls:
    def __init__(self):
        pass
    def demoFunc0(self):
        pass
    def demoFunc1(self, arg1):
        pass

def RunFuzzer (x):
    pass
```

Procedure-oriented

```
class demoCls:
    def __init__(self):
        pass
    def demoFunc0(self):
        pass
    def demoFunc1(self, arg1):
        pass

def RunFuzzer (x):
    pass
```

Program structure

```
class demoCls (PY_LIB.CLS):
    def __init__(self):
        pass
    def CLS_func0(self):
        pass
    def CLS_func1(self, arg1):
        pass

def RunFuzzer (x):
    pass
```

```
For intV in range (x, y):
    pass
```

```
For obj in iterObj:
    pass
```

Function structure

```
if x:
    pass
```

```
if x:
    pass
Else:
    pass
```

ScriptL: a simple script language for describing the semantic of python APP generation.

ScriptL Interpreter: interpreter the scripts and generate new python applications for various run-time APIs and python built-in functions.

[Command]: newOO <API-name>

[Description]: generate a new object-oriented program for the API “API-name”

[Command]: newPO <API-name>

[Description]: generate a new procedure-oriented program for the API “API-name”

[Command]: newInherit <API-name>

[Description]: Inherit the class of “API-name” and re-implement “API-name”

[Command]: FOR N <API-name>

[Description]: Wrap the API “API-name” into a for loop with N iterations

[Command]: Recurve <API-name>

[Description]: Recurve invoking the API “API-name” into a for loop with N iterations

[Command]: Randcall <API-name>

[Description]: Call the API “API-name” with random APIs in the corpus

[Command]: OOcall <API-name>

[Description]: Call the API “API-name” in out-of-order with its dependences

