U.S. Department of Energy

Office of Science

Zolera Soap Infrastructure (ZSI)

Keith R. Jackson

Lawrence Berkeley National Laboratory

Overview

- Python Web Service Support
- ZSI Features
- Recent Improvements
- Using ZSI
- Examples
- Future Work
- Conclusion

Python Web Service Support

- SOAP.py and ZSI are two of the Web Service toolkits for python
  - Both support RPC style SOAP messaging
  - ZSI supports document style SOAP messaging
  - Both have basic WSDL support
    - Share the same code base for this
  - Both rely on PyXML and 4Suite for their underlying XML support

Python Web Service Support

- SOAP.py is easier to use for simple Web Services
- ZSI is more complicated, but provides more flexibility to the programmer
- We began using ZSI for our work mainly due to our need for literal encoding
- Project URL for both:
  - http://pywebsvcs.sf.net/

ZSI Features

- Automatic code generation from WSDL and XML Schema
  - Can generate both client stubs and server skeletons
  - Supports mapping XML Schema types to python types
- XML Security
  - Implemented XML Digital Signature and XML Encryption
    - Based on PyOpenSSL

ZSI Features

- Web service hosting environment
  - Based on the new Twisted 2.0 release
  - Container handles most common functionality
    - Security, dispatch, etc.
- Lightweight stand-alone web service support
  - Based on the BaseHTTPServer class
  - Tradeoff between performance and size

Recent Improvements

- Currently two branches in CVS:
  - The HEAD is the old code base. Good support for:
    - rpc/encoded
  - serialize-dom-scheme branch is the new development branch
    - Changed the way the output messages are constructed to support WS-Security
    - Focus on WS-I BP-1.x compliance
      - document/literal support initially

Recent Improvements

- Extended code generation support
  - Client calls look like method invocations
  - Servers can be implemented by:
    - Sub-classing an automatically generated class
    - Passing an implementation class into the constructor
- Standard authorization callback
  - Container calls out before invoking any operations
  - Standard SOAP context object

Recent Improvements

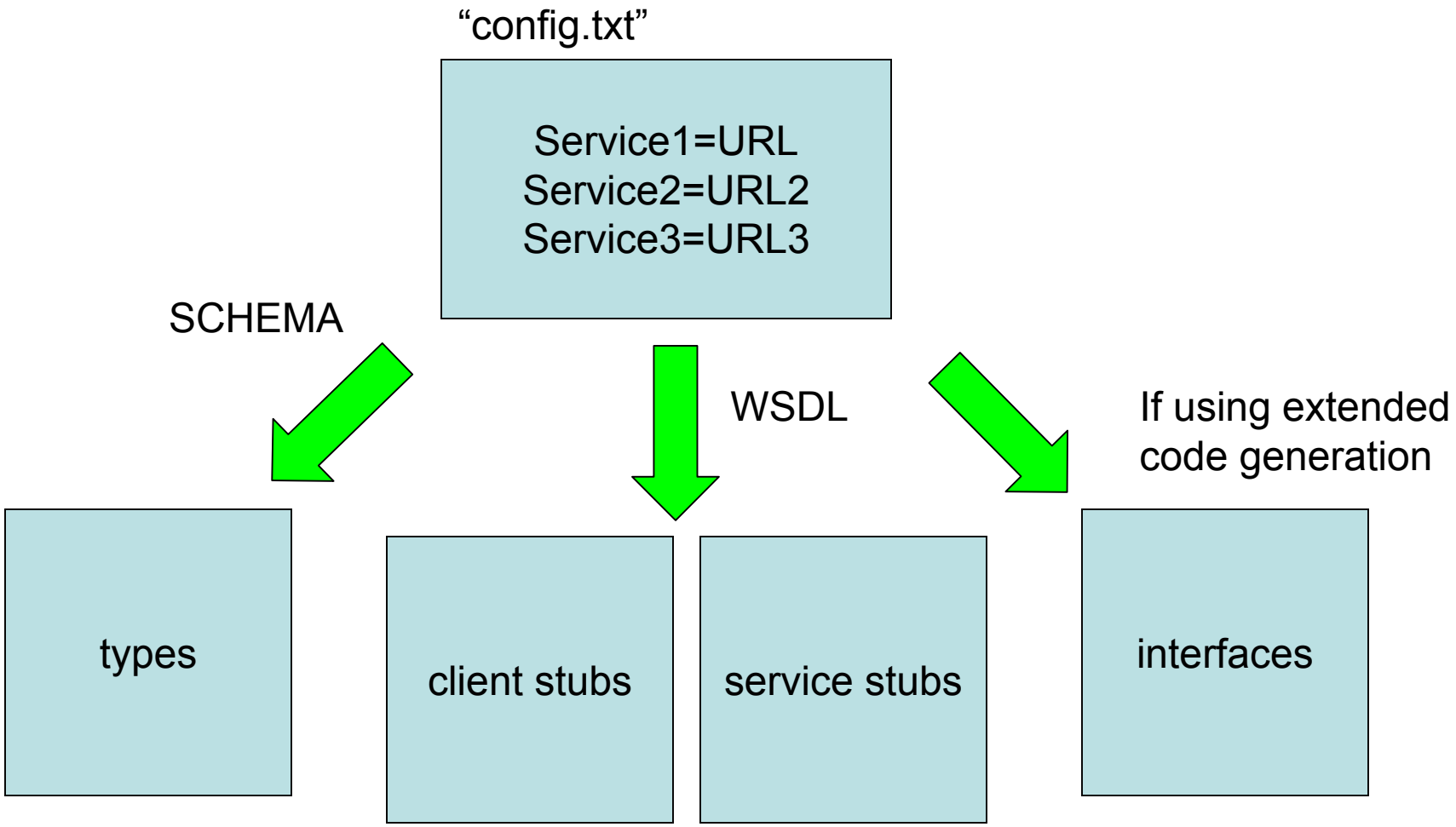
- Performance
  - Factor of 20 improvement switching to 4Suite's cDomlette for XML Parsing
    - Recent 1.0b2 release saw a 10% improvement
  - Twisted 2.0 improved the underlying http performance
  - Implemented SSL session caching
    - Large improvement by avoiding the SSL handshake protocol

Using ZSI

- Can either invoke wsdl2py by hand, or use the setup.py script to generate bindings from a WSDL file
  - If using distutils, config.txt contains the url for the WSDL document
- Must be able to resolve all WSDL and XML Schema documents
- Need:
  - PyXML, 4Suite, PyOpenSSL for security, and Twisted for container support

Using ZSI

%python setup.py --regenerate build



Usage Example


```
from Registration_services import *

locator = RegistrationServiceLocator()
service = locator.getRegistration(
                                  tracefile=sys.stdout)
request = GetUserRequestWrapper()
request._UserId = "john_doe"
response = service.GetUser(request)
print '  Age: %d' % response._User._Age
print ' Name: %s' % response._User._Name
print 'UserId: %s' % response._User._UserId
```

Usage Example


```
from Registration_services import *

def GetUser(user_id):
    user = ns1.User_Def()
    user._UserId = user_id
    user._Name = "John Doe"
    user._Age = 32

    response = GetUserResponseWrapper()
    response._User = user
    return response
```

New Example

```
from EchoServer_client import
     EchoServerSOAP

hw = EchoServerSOAP(
"http://localhost:9999/EchoServer")

print hw.Echo("Test TIE String")
```

New Example


```
class EchoServIn(EchoServer):
    def __init__(self, post='', **kw):          EchoServer.__init__(
                                                self, post, kw=kw)
    def Echo(self, input):
     return input * 3
serviceContainer =
        ServiceContainer(('localhost', 9999))
hws2 = EchoServIn()
serviceContainer.setNode(hws2,
                         post='/EchoServIn')
serviceContainer.serve_forever()
```

Future Plans

- Merge the two branches
  - Add rpc/encoded support into the SDS branch
    - Only two tests left that fail
  - Add extended code generation support into SDS branch
- BP-1.1 compliance
  - Implement SOAP with attachments
- SOAP 1.2 and WSDL 2.0

Future Plans

- Performance
  - HTTP 1.1 persistent connections
  - Alternate XML parser implementations
    - cElementTree
- Improved programming model
  - Improve ease-of-use for programmers
    - Python annotations for declarative programming
  - Simplicity versus flexibility

Conclusion

- The Python Web Service work is funded by the U.S. Department of Energy Office of Science
- ZSI
  - http://pywebsvcs.sourceforge.net/zsi.html
- Twisted
  - http://twistedmatrix.com/products/twisted
- 4Suite
  - http://4suite.org/index.xhtml
- Email
  - KRJackson@lbl.gov