monadWS: A Monad-Based Testing Tool for Web Services

College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, China

Yingzhou Zhang, Wei Fu, Changhai Nie

Email: zhangyz@njupt.edu.cn
Agenda

• About monadWS
• Monads Techniques
• The monads in monadWS
  – HSP monad
  – Gen monad
  – TestM monad
• Tool Snapshots
• Conclusion
About *monadWS*

- An automatic WS testing tool based on monads techniques

*monadWS* adopts

- **HSP monad**\(^{[1]}\) for describing WS test cases
- **Gen monad**\(^{[2]}\) for generating test data automatically, and
- **TestM monad** for building the whole WS testing.

---


Its framework

- WSDL Specification
- Operations
- XML Schema
- SoapActions
- Services
- Data Generation Strategy
- SOAP Generator
- test cases (XML files)
- SOAP Caller
- QuickCheck Properties
- Test Reports
Agenda

• About *monadWS*
• Monads Techniques
• The monads in *monadWS*
  – HSP monad
  – Gen monad
  – TestM monad
• Tool Snapshots
• Conclusion
Monads Intro.

- A general functional notation
  \[ f : a \rightarrow b \]
- Its monadic version of this notion
  \[ f : a \rightarrow M b \]
- Operations on monad \( M \)
  \[ \text{return} :: a \rightarrow M a \]
  \[ \text{bind} :: M a \rightarrow (a \rightarrow M b) \rightarrow M b \]

Therefore, a monad can be thought as a strategy for combining computations into more complex computations.
Agenda

• About *MonadWS*
• Monads Techniques
• The monads in *MonadWS*
  – HSP monad
  – Gen monad
  – TestM monad
• Tool Snapshots
• Conclusion
HSP Monad: represent TC

◆ represents a WS test case as a SOAP message
◆ can encapsulate any side effects, and lets all XML expressions be computed safely

\[
\text{makeTC}_\text{serv} :: \text{InputType} \rightarrow \text{HSP XML}
\]

The input type of a service operation (say \text{serv}), such as String, Double …
HSP Monad: represent TC

\[
\text{makeTC\_serv} :: \text{InputType} \rightarrow \text{HSP XML}
\]

For example:

```haskell
makeTC\_getOffsetUTCTime :: \text{Double} \rightarrow \text{HSP XML}
makeTC\_getOffsetUTCTime \text{\_d} =

\[
<\text{soap:Envelope xmlns:soap=soapURI xmlns:xsd=xsdURI}>
<\text{soap:Body}>
<\text{getOffsetUTCTime xmlns=operURL}>
<\text{hoursOffset} \%
\text{show} \text{\_d} \%
</\text{hoursOffset}>
</\text{getOffsetUTCTime}>
</\text{soap:Body}>
</\text{soap:Envelope}>
\]

\text{where}

\text{soapURI} = "http://schemas.xmlsoap.org/soap/envelope/" 
\text{xsdURI} = "http://www.w3.org/2001/XMLSchema"
\text{operURL} = "http://www.Nanonull.com/TimeService/"
```
HSP Monad: represent TC

```haskell
makeTC_getOffsetUTCTime :: Double -> HSP XML
makeTC_getOffsetUTCTime d =
  <soap:Envelope xmlns:soap=soapURI xmlns:xsd=xsdURI>
    <soap:Body>
      <getOffsetUTCTime xmlns=operURL>
        <hoursOffset> <\% show d \%> </hoursOffset>
      </getOffsetUTCTime>
    </soap:Body>
  </soap:Envelope>
where
  soapURI = "http://schemas.xmlsoap.org/soap/envelope/"
  xsdURI = "http://www.w3.org/2001/XMLSchema"
```

HSP provides interfaces for other computations through the escapes (with “<%” and “%>”)

Gen Monad: generate data

- automatically generates data for simple types (such as Double, Int, String and so on)

```
class Arbitrary a where
    arbitrary :: Gen a
```

```
instance Arbitrary String ...
instance Arbitrary Double ...
```
Gen Monad: generate data

\[
\text{class Arbitrary a where}
\]

\[
\text{arbitrary :: Gen a}
\]

\[
\text{makeTC_getOffsetUTCTime :: Double \to HSP XML}
\]

- **general testing method:**
  - Step 1. generate a test value
  - Step 2. generate the final test case with the value

\[
\text{testTCGen :: IO String}
\]

\[
\text{testTCGen = do}
\]

\[
\text{d <- generate (arbitrary :: Gen Double)}
\]

\[
\text{return \$ hsp2str (makeTC_getOffsetUTCTime d)}
\]
Gen Monad: generate data

\[
\text{makeTC\_getOffsetUTCTime} :: \text{Double} \rightarrow \text{HSP XML}
\]

- **general testing method:**

  \[
  \text{testTCGen} :: \text{IO} \text{ String}
  \]
  \[
  \text{testTCGen} = \text{do}
  \]
  \[
  d \leftarrow \text{generate} \ (\text{arbitrary} :: \text{Gen Double})
  \]
  \[
  \text{return} \ \$ \ \text{hsp2str} \ (\text{makeTC\_getOffsetUTCTime} \ d)
  \]

- **monadic method by liftM: on-the-fly generate**

  \[
  \text{testTCGen2} :: \text{IO} \text{ String}
  \]
  \[
  \text{testTCGen2} =
  \]
  \[
  \text{liftM} \ (\text{hsp2str} \ . \ \text{makeTC\_getOffsetUTCTime}) \ \$ \ \text{generate} \ (\text{arbitrary} :: \text{Gen Double})
  \]
Gen Monad: generate data

- We can easily change the generation strategy through the instances of Arbitrary class.

```haskell
class Arbitrary a where
    arbitrary :: Gen a

instance Arbitrary Double where
    arbitrary = myDoubleGen

myDoubleGen :: Gen Double
    myStringGen = suchThat arbitrary guard
    where
        guard a = (a > 0) && (a < 1000)
        || elem a [-1, 0, 9999]
```

```haskell
arbitrary = myDoubleGen
```
TestM Monad: testing as a monad

- processes WS testing as the following TestM monad:

```haskell
type TestM a = CoAlgMonadT (EnvT TestEnv (StateT TestState HSP)) a
```

- **Coalgebraic monad** for WS computing
- **Environment monad** for testing environment (e.g. Schema type information, test cases files)
- **State monad** for testing states (e.g. executing time, testing results)
CPU Time Results of Autogeneration Test Cases for TimeService

Execution Time of the SOAP Messages of getOffesetUTCTime in TimeService
## Comparison of WSTester with Other Test Tools for Web Services

<table>
<thead>
<tr>
<th>websites</th>
<th>WSTester</th>
<th>soapUI</th>
<th>SoapTest</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>oper</td>
<td>Time (sec.)</td>
<td>fail</td>
</tr>
<tr>
<td><a href="http://www.webservice.net/country.asmx">http://www.webservice.net/country.asmx</a></td>
<td>10</td>
<td>0.406</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www.webservice.com/globalweather.asmx">http://www.webservice.com/globalweather.asmx</a></td>
<td>2</td>
<td>0.078</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www.webservice.net/CurrencyConverter.asmx">http://www.webservice.net/CurrencyConverter.asmx</a></td>
<td>1</td>
<td>0.078</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www.webservice.com/airport.asmx">http://www.webservice.com/airport.asmx</a></td>
<td>4</td>
<td>0.109</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www.webservice.net/BibleWebservice.asmx">http://www.webservice.net/BibleWebservice.asmx</a></td>
<td>4</td>
<td>0.265</td>
<td>3</td>
</tr>
<tr>
<td><a href="http://www.webservice.net/Statistics.asmx">http://www.webservice.net/Statistics.asmx</a></td>
<td>1</td>
<td>0.125</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www.xignite.com/xcompensation.asmx">http://www.xignite.com/xcompensation.asmx</a></td>
<td>10</td>
<td>0.203</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www.webxml.com.cn/WebServices/WeatherWebService.asmx">http://www.webxml.com.cn/WebServices/WeatherWebService.asmx</a></td>
<td>5</td>
<td>0.172</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www.webservice.net/sendsmsworld.asmx">http://www.webservice.net/sendsmsworld.asmx</a></td>
<td>1</td>
<td>0.047</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://www.webservice.net/stockquote.asmx">http://www.webservice.net/stockquote.asmx</a></td>
<td>1</td>
<td>0.031</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://fy.webxml.com.cn/webservices/EnglishChinese.asmx">http://fy.webxml.com.cn/webservices/EnglishChinese.asmx</a></td>
<td>6</td>
<td>0.094</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://webservice.webxml.com.cn/WebServices/MobileCodeWS.asmx">http://webservice.webxml.com.cn/WebServices/MobileCodeWS.asmx</a></td>
<td>2</td>
<td>0.047</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://webservice.webxml.com.cn/WebServices/WeatherWS.asmx">http://webservice.webxml.com.cn/WebServices/WeatherWS.asmx</a></td>
<td>6</td>
<td>0.140</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://webservice.webxml.com.cn/WebServices/StockInfoWS.asmx">http://webservice.webxml.com.cn/WebServices/StockInfoWS.asmx</a></td>
<td>1</td>
<td>0.078</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://webservice.webxml.com.cn/WebServices/ChinaOpenFundWS.asmx">http://webservice.webxml.com.cn/WebServices/ChinaOpenFundWS.asmx</a></td>
<td>6</td>
<td>0.328</td>
<td>0</td>
</tr>
<tr>
<td><a href="http://webservice.webxml.com.cn/webservices/DomesticAirline.asmx">http://webservice.webxml.com.cn/webservices/DomesticAirline.asmx</a></td>
<td>2</td>
<td>0.109</td>
<td>0</td>
</tr>
</tbody>
</table>
Agenda

• About monadWS
• Monads Techniques
• The monads in monadWS
  – HSP monad
  – Gen monad
  – TestM monad
• Tool Snapshots
• Conclusion
Snapshots of our tool *monadWS*
Snapshots of our tool *monadWS*
Snapshots of our tool \textit{monadWS}
Agenda

• About monadWS
• Monads Techniques
• The monads in monadWS
  – HSP monad
  – Gen monad
  – TestM monad
• Tool Snapshots
• Conclusion
Conclusion

• briefly introduce our monad-based testing tool, *MonadWS*
• To automatically test web services.
• *MonadWS* can use monads related to help
  – WS test case representation,
  – test data autogeneration, and
  – test auto-execution.
Conclusion

• *monadWS* can use monads related to help
  – WS test case representation,
  – test data autogeneration, and
  – test auto-execution.

• Future work
  – More comparisons with existing tools
  – Use more data-generation strategy
  – Apply to WS composition (with BPEL)
Thank you!