Generating Test Cases for Web Services Using Data Perturbation

Wuzhi XU

wxu2@gmu.edu

Department of Information and Software Engineering

George Mason University

Fairfax, VA 22030-4444 USA

Joint Research with Dr. Jeff Offutt

Supported by NASA

Introduction

- Black box testing of web services
- Focus on peer to peer interactions
- Data perturbation of existing XML message
- Based on XML schema & type rules

Web Services

- *Internet-based, modular applications that perform a specific business task and conform to a particular technical format. – IBM*

- An Internet-based, modular application that uses Simple Object Access Protocol for communication and transfers data in XML through the Internet.

Technologies in Web Services

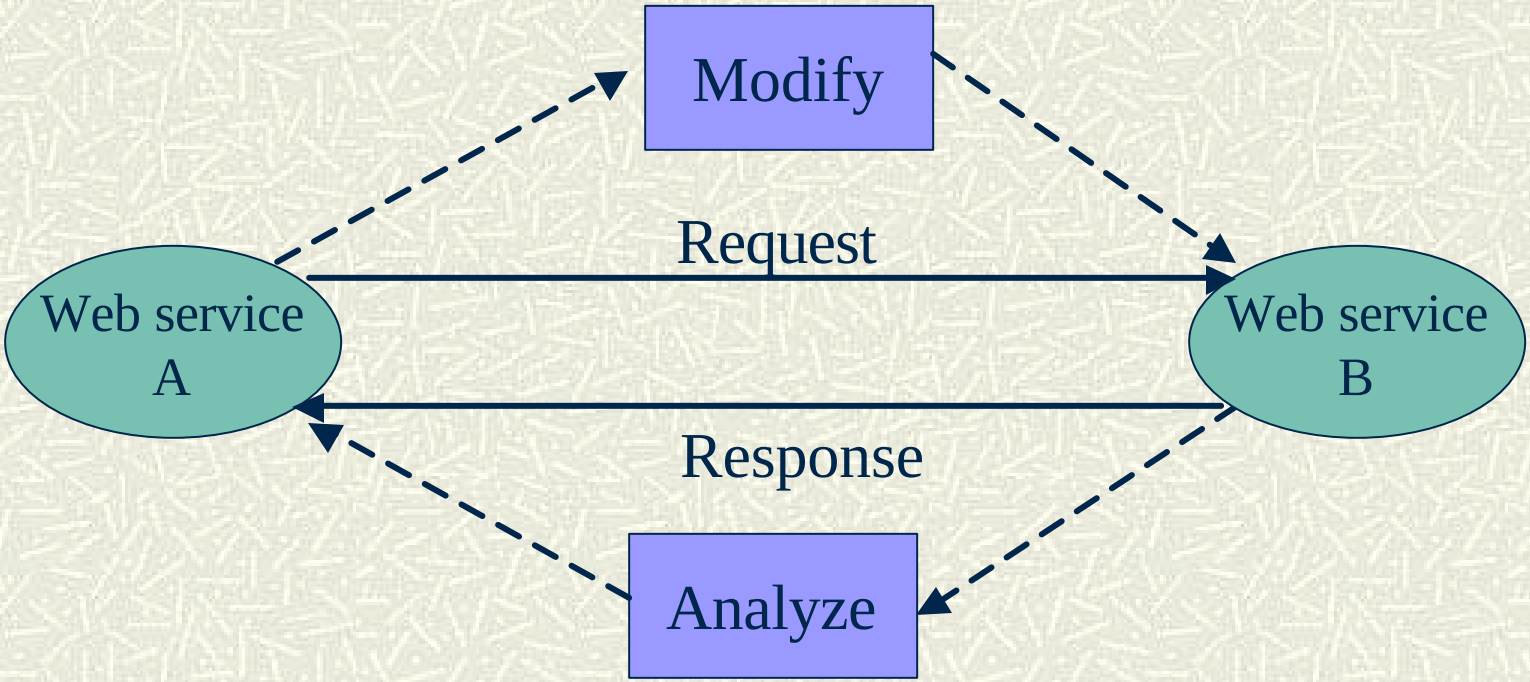
- Common communication infrastructure
  - Widely supported
  - Internet-based
  - Implements standardized technical formats
- Advantages of this technology
  - Independent of hardware / software platform
  - Reduces the complexities and cost of software integration
- Web services technologies
  - Extensible Markup Language (XML)
  - Universal Description, Discovery and Integration (UDDI)
  - Web Services Description Language (WSDL)
  - Simple Object Access Protocol (SOAP)

Testing Web Services

- Challenges of testing web services
  - Will web services interactions be handled acceptably well?
  - Will requests be handled for all types of data?
- Characteristics with testing web services
  - No access to source
  - Low testability
  - Dynamic and loosely coupled
  - Heterogeneous
  - Independent platforms
  - No UI
- Testing capabilities involved in testing web services
  - Testing SOAP messages
  - Testing WSDL files and using WSDL files to generate test plans
  - Testing the publish, find, and bind capabilities of web services

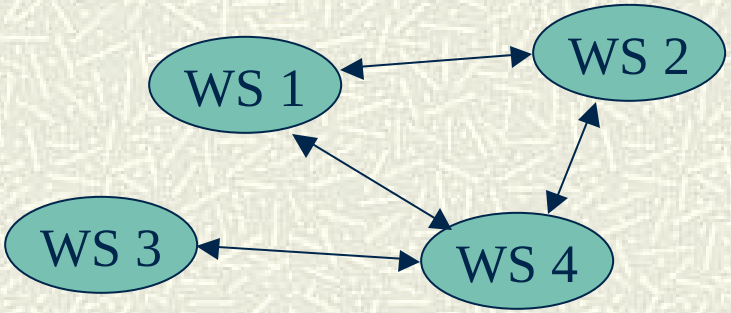
Our Solution

1. Modify the request messages
2. Send the modified messages
3. Analyze the responses



Solution Characterization

- Formal model for XML
- Rules for perturbing values
- Peer-to-peer, not multilateral
- Testing SOAP messages
- Black box testing



Multilateral Web Services Interaction

Peer-to-peer Web services interaction

The Formal Model for XML Schema

- The Regular Tree Grammar (RTG) is a 6-tuple

  <E, D, N, A, P, n_s >

- XML message and schema for the books example

- The simplified XML message

```
<books>
 <book>
   <ISBN>0-672-32374-5</ISBN>
   <price>59.99</price>
   <year>2002</year>
 </book>
 <book>
   <ISBN>0-781-44371-2</ISBN>
   <price>69.99</price>
   <year>2003</year>
 </book>
</books>
```

- The simplified XML schema

```
<xs:element name="books">
<xs:complexType>
  <xs:sequence>
    <xs:element name="book“
     maxOccurs="unbounded”>
     <xs:complexType>
       <xs:sequence>
         <xs:element name="ISBN" type="xs:string"/>
         <xs:element name="price" type="xs:double"/>
         <xs:element name="year" type="xs:int"/>
       </xs:sequence></xs:complexType></xs:element>
</xs:sequence></xs:complexType>  </xs:element>
```

The RTG Model for the example

- The RTG for the books

E={**books**, **book**, **ISBN**, **price**, **year**}

D={**string**, **double**, **int**}

N={n_s, n_b, n_i, n_r, n_y }

P={n_s --> **books** <(n_b)*>,

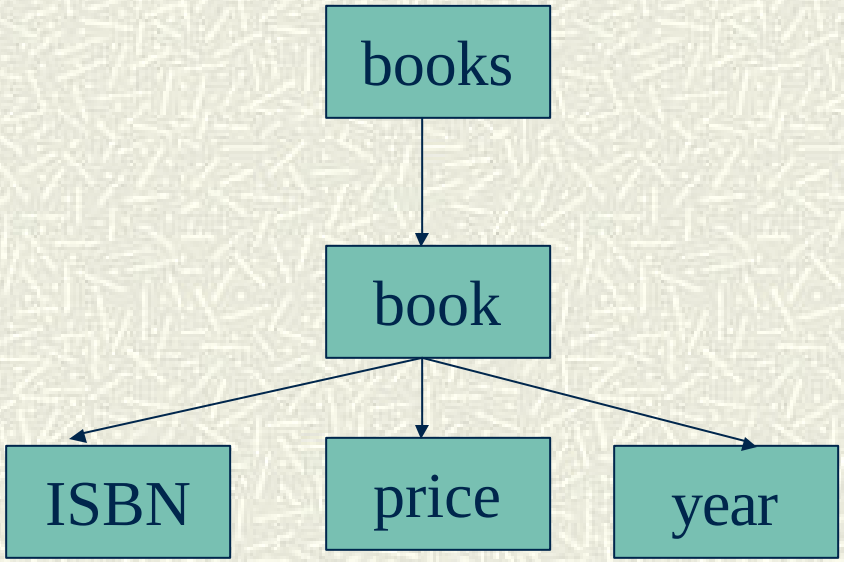
n_b --> **book** < n_i n_r n_y>,

n_i --> **ISBN** <**string**>,

n_r -->**price** <**double**>,

n_y --> **year** <**int**> }

- The derived tree for the books



Our Approach

- Data Perturbation (DP): modify the request messages and analyze the response messages
- Two steps in data perturbation:
  1. Data value perturbation: modifying values in messages based on their data type
  2. Interaction perturbation: Interaction is classified into RPC communication and data communication
     1. RPC communication perturbation
     2. Data communication perturbation

Data Value Perturbation (DVP)

- Three data types are currently considered: string, numeric, and enumeration
- Boundary value test cases for these data types:
  - String: maximum length, minimum length, zero length, upper case, lower case
  - Numeric: maximum value, minimum value, zero
  - Enumeration: boolean set, finite set of values
- One example for numeric data type:

Original value	Perturbed value	Test case
<price>59.99</price>	2^{63}-1	Maximum value
	- 2^{63}	Minimum value
	0	Zero

RPC Communication Perturbation (RCP)

- Test cases are generated in term of data uses
- Two kinds of data uses: normal use and SQL use
- Mutation analysis is extended to testing normal uses
- Traditional mutation operators are redefined
- SQL injection is extended to testing SQL uses

RCP Example

- **Message before modification**

```
<adminLogin>
    <arg0 xsi:type="xsd:string">turing</arg0>
    <arg1 xsi:type="xsd:string">enigma</arg1>
</adminLogin>
```

- **Message after modification**

```
<adminLogin>
    <arg0 xsi:type="xsd:string">turing' OR
    '1'='1</arg0>
    <arg1 xsi:type="xsd:string">enigma' OR
    '1'='1</arg1>
</adminLogin>
```

- **Resulting SQL query**

```
SELECT username FROM adminuser WHERE
        username='turing'
    AND password='enigma'
```

- **Modified SQL query**

```
SELECT username FROM adminuser WHERE
        username='turing'
    OR '1'='1'
    AND password='enigma'
    OR '1'='1'
```

Data Communication Perturbation (DCP)

- Test relationships and constraints in data communication messages

- Definitions of relationship and constraint:
  - Relationship is the parent-child association between two non-terminal elements

    It reflects the cardinality constraints in a XML schema
  - Constraint is the association between a non-terminal element and a terminal element

    It reflects the expression of element values

DCP Example

- **Duplicate one book instance**

```
<books>
  <book>
    <ISBN>0-672-32374-5</ISBN>
    <price>59.99</price>
    <year>2002</year>
  </book>
  <book>
    <ISBN>0-781-44371-2</ISBN>
    <price>69.99</price>
    <year>2003</year>
  </book>
  <book>
    <ISBN>0-781-44371-2</ISBN>
    <price>69.99</price>
    <year>2003</year>
  </book>
</books>
```

- **Delete one book instance**

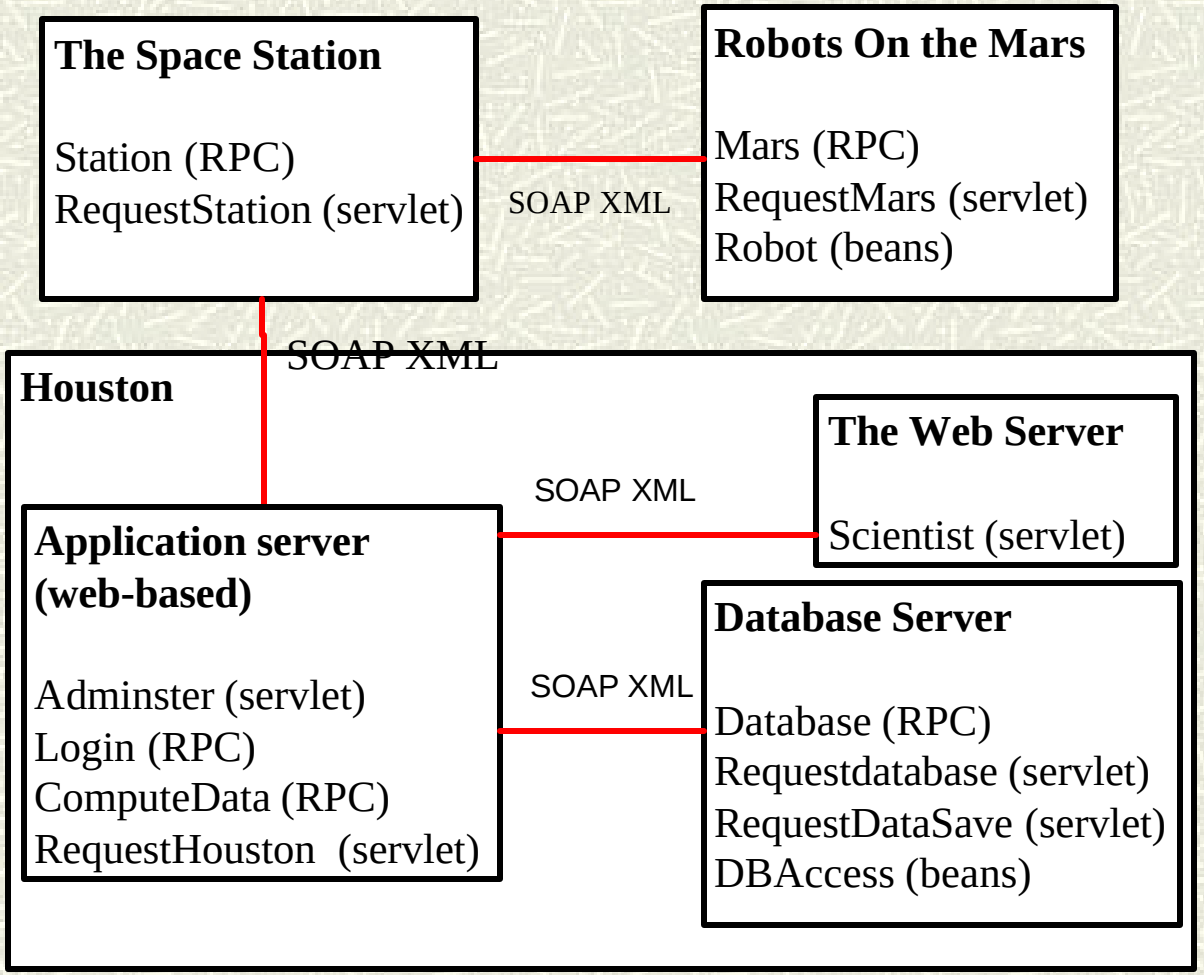
```
<books>
  <book>
    <ISBN>0-672-32374-5</ISBN>
    <price>59.99</price>
    <year>2002</year>
  </book>
</books>
```

Preliminary Case Study

- Used web services developed for the study
- Inserted faults by hand
- Evaluated tests in terms of numbers of faults found

The Web Services for the Experiment

- The Mars Robot Communication System
  - Five web services
  - Six servlets
  - Two Java beans



Faults and Test Results

- 18 faults were inserted into the application
- 14 seeded faults found plus two natural faults found

	Number of faults
Predicate faults	5
Computation faults	3
SQL faults	3
Wrong method call	3
XML and SOAP faults	2
Other faults	2

	DVP	RCP	DCP	**Total**
Number of test cases	100	15	27	**142**
Faults found (seeded)	14	7	4	**14**
Faults found (natural)	1	1	0	**2**

Conclusion

- Effective and practical way to generate tests for web services without requiring access to the source
- RPC communication perturbation: syntax-based modification on values in messages
- Data communication perturbation:
  - Little work on testing data communication message
  - Focus on testing the semantics
- Future work:
  - Relative usefulness of DVP, RCP, and DCP
  - Testing those applications that need no data inputs
  - Multilateral communication
  - Determining output correctness