Towards Automated Testing of Web Service Choreographies

Felipe Besson, Pedro Leal, Fabio Kon and Alfredo Goldman
{besson, pedrombl, fabio.kon, gold}@ime.usp.br
Department of Computer Science - IME - University of São Paulo
Deja Milojicic - dejan@hpl.hp.com
Hewlett Packard Laboratories

6th International Workshop on Automated Software Testing

Introduction

Web service choreographies have been proposed as a **decentralized scalable way** of composing services.

Inherent characteristics of Service-Oriented Architecture (SOA) such as **dynamicity**, **third-party** and **governance issues** and the **decentralized flow of information**, makes the automated testing of choreographies difficult.

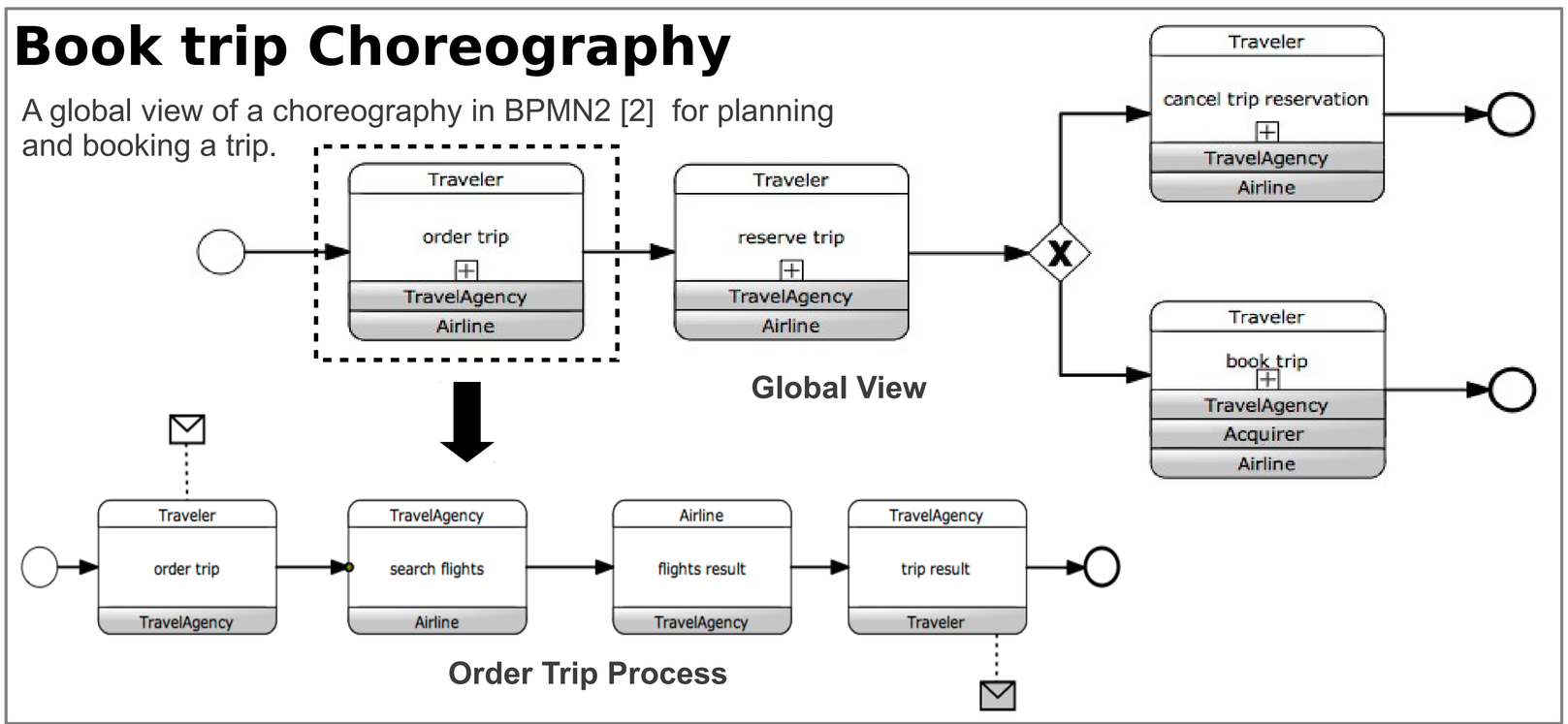
Goals

We aim to develop a testing framework for supporting Test-Driven Development (TDD) [1] of choreographies. During the development, the framework will provide features for automated testing of:

- Isolated services;
- Messages exchanged in the choreography;
- The entire choreography.

Book trip Choreography

A global view of a choreography in BPMN2 [2] for planning and booking a trip.



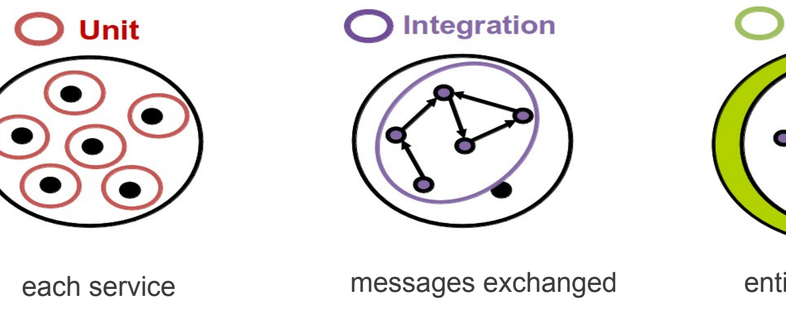
Our Prototype

This work represents our first efforts to achieve our goals. Our prototype consists of:

- *Ad hoc* bash scripts for the Book Trip **choreography enactment** (using OpenKnowledge [3]);
- A set of JUnit test cases for **automated testing** of this choreography (see the tests below);

Acceptance Testing

The choreography is tested from the user perspective. In this context, the choreography is accessible as an atomic service, and each test exercises an entire conversation as a unit.

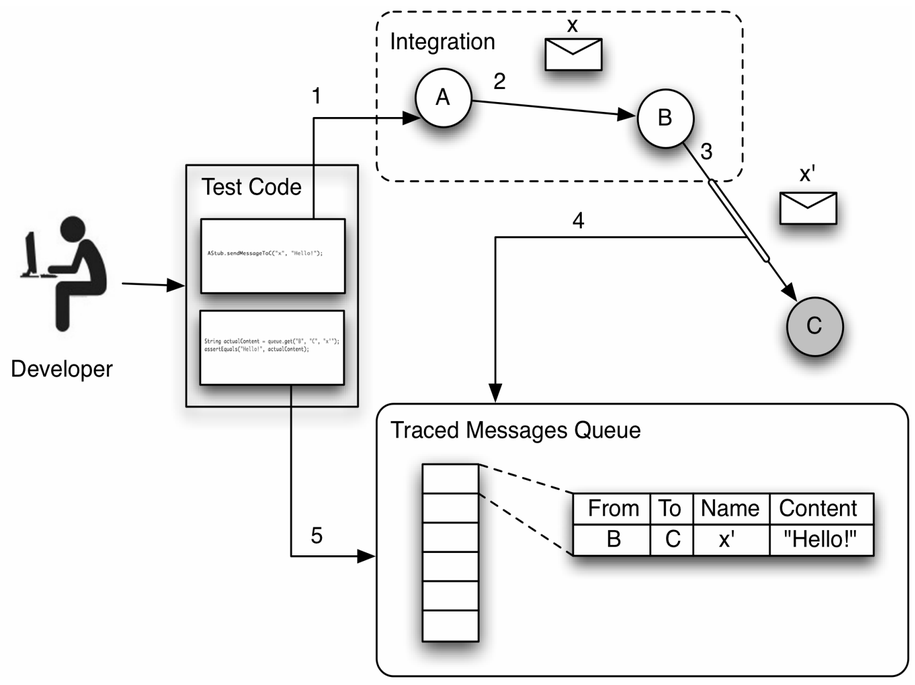


each service | messages exchanged | entire choreography

The steps of testing

Integration Testing

1. Developer activates the choreography, invoking service A;
2. Messages are exchanged between services A and B;
3. Output messages from B are intercepted;
4. Messages are stored in a queue;
5. Collected data is validated.



Unit Testing

Every operation of each service participating in the choreography is tested:

```
private class AirlineWSTest  {

 private AirlineWSService service;
 private AirlineWS stub:

 @BeforeClass
  public static void publishService() {
    Bash.deployService ("airline"); }
 ...
 @Test
 public void shouldFindFlight() {
    flight = airlineWS.getFlight ("Honolulu", ... );
    assertEquals ("3153", fligth.getId());
    assertEquals ("Honolulu", flight.getDestination());
    assertEquals ("05-21-11", flight.getDate());
    assertEquals ("09:15", flight.getTime());
 } ...
```

SOAP service

```
private class TravelAgencyWSTest  { ...

 BASE_URL = "http://localhost:9881/travelagency";
 private static RESTClient client;

 @BeforeClass
  public static void publishService() {
    Bash.deployService ("travelagency");
    client = new RESTClient();
    client.setBaseURL(BASE_URL);}
 ...
 @Test
  public void shouldRetriveCreditCardNumber() {
    body = "John|421543-2";
    client.POST("/users", body);
    response= client.GET("/users?name=John");
    assertEquals ("421543-2", response);
   } ...
```

REST service

Code example

This code illustrates a simple interface to validate a message exchanged by the Airline and Travel Agency services.

```
travelerStub.bookReserve(reserve);

String expected = queue.get("airline",
                            "travelagency",
                            "value_paid");

assertEquals("BRL 2000", expected);
```

Ongoing work

We are extending our prototype by providing features for:

- Generating web service clients dynamically;
- Manipulating the elements (roles, messages, services) of a choreography more easily;
- Mocking third-party services and choreography parts;
- Improving the interception and validation of exchanged messages.

More information: http://ccsl.ime.usp.br/baile/VandV

References

[1] Kent Beck. Test Driven Development: By Example. Addison-Wesley Professional, 2002.
[2] OMG. Business Process Model and Notation. Available on: <http://www.bpmn.org/>.
[3] OpenKnowledge Project. Available on: <http://www.openk.org/>

Acknowledgments

This research is funded by:

EUROPEAN COMMISSION

CHOReOS
Large Scale Choreographies for the Future Internet