แนวทางในการทดสอบฟังก์ชั่นการทำงานของเว็บเซอร์วิสแบบ REST

REST Web Services Functional Testing Guidelines

กานดา รุณนะพงศา[1], พงษ์ศักดิ์ ฮุนตระกูล[2], เสกสิทธิ์ สุวรรณ[3], ชัยวัฒน์ บุตรไชย[1], ภาณุวัฒน์ คามวัลย์[4]

[1]ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยขอนแก่น จ. ขอนแก่น

[2]สถาบันศศินทร์ จุฬาลงกรณ์มหาวิทยาลัย กรุงเทพ, [3] ฝ่ายวิจัยและพัฒนาเทคโนโลยีคอมพิวเตอร์เพื่อการคำนวณ ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ จ. ปทุมธานี,

[4]สำนักส่งเสริมเครือข่ายวิสาหกิจคอมพิวเตอร์ สำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติ จ.ปทุมธานี

Email: krunapon@kku.ac.th, pongsak@hoontrakul.com, somyos.ak@morethailand.com, seksit@gmail.com, chaiwat.bootchai@gmail.com, panuwat.khamwan@gmail.com

บทคัดย่อ

เว็บเซอร์วิส (Web Service) เป็นระบบซอฟต์แวร์ที่ออกแบบมาเพื่อสนับสนุนการทำงานระหว่างคอมพิวเตอร์กับคอมพิวเตอร์ผ่านระบบเครือข่าย โดยใช้ภาษาเอ็กซ์เอ็มแอล (XML) เป็นภาษาในการติดต่อสื่อสารระหว่างโปรแกรม ปัจจุบันมีการพัฒนาเว็บเซอร์วิสทั้งแบบ SOAP (Simple Object Access Protocol) และ REST (Representational State Transfer) ถึงแม้ว่าเว็บเซอร์วิสแบบ SOAP จะเป็นมาตรฐานกลางซึ่งเกิดขึ้นมาก่อนเว็บเซอร์วิสแบบ REST แต่ปัจจุบันมีการพัฒนาและเรียกใช้เว็บเซอร์วิสแบบ REST มากขึ้นเนื่องจากเว็บเซอร์วิสแบบ REST นั้นพัฒนาและเรียกใช้ง่ายกว่าแบบ SOAP แต่ทั้งนี้ทั้งนั้นไม่ว่าผู้พัฒนาเว็บเซอร์วิสจะพัฒนาเว็บเซอร์วิสแบบใด ผู้พัฒนาเว็บเซอร์วิสจำเป็นต้องทดสอบเว็บเซอร์วิสก่อนที่จะเปิดบริการเว็บเซอร์วิสให้ผู้อื่นเรียกใช้ บทความนี้นำเสนอแนวทางในการทดสอบความสมบูรณ์และความถูกต้องของฟังก์ชั่นการทำงานของโอเปอเรชั่นต่าง ๆ ของเว็บเซอร์วิสแบบ REST ซึ่งแนวทางในการทดสอบนี้สามารถนำไปทดสอบกับเว็บเซอร์วิสแบบ REST ใด ๆ โดยที่เท่าที่ทราบยังไม่มีบทความใด ๆ นำเสนอวิธีทดสอบฟังก์ชั่นการทำงานของโอเปอเรชั่นต่าง ๆ ของเว็บเซอร์วิสแบบ REST

คำสำคัญ: *เว็บเซอร์วิส, REST, การทดสอบ, คุณภาพของเซอร์วิส*

Abstract

A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. XML is used as the language for describing software interface and communicating between programs. Currently, Developing Web Services is taken into two approaches: SOAP (Simple Object Access Protocol) and REST (Representational State Transfer) Although SOAP Web Services are the standard offered before REST Web Services, REST Web Services have become popular due to its more simplicity in developing and calling. No matter what type of Web services is, Web service developers need to test their Web services before deploying and providing services for others to use. This article proposes the guideline in testing the completion and correctness of functionality of operations in REST Web Services. This test guideline can be applied for testing the functionality of any REST Web Service. To the best of our knowledge, there hasn't been any article proposes such guideline.

Key Words: *Web Services, REST, Testing, Quality of Service*

[π] Without implying explicitly or implicitly that they agree with or are responsible for the partial or whole contents of the study, our special thanks are for Mr. Somyos Akahadsri (somyos.ak@morethailand.com) of A2Z Professional Trade Co., Ltd. (www.A2Zprotravel.com) Bangkok and Mr. Tone Tansuitiwong.

1. บทนำ

อินเตอร์เน็ตได้ทำให้การสื่อสารข้อมูลระหว่างมนุษย์เป็นไปได้อย่างสะดวกและรวดเร็ว ซึ่งได้มีผลกระทบต่อหลายวงการ อย่างเช่น ในการศึกษานั้น ทำให้ผู้เรียนสามารถศึกษาสิ่งต่าง ๆ ได้จากสื่ออิเล็กทรอนิกส์ซึ่งสามารถเข้าถึงได้ผ่านอินเตอร์เน็ต ในการทำธุรกิจ ทำให้เจ้าของผลิตภัณฑ์ พันธมิตรคู่ค้า ตลอดจนลูกค้า มีการสื่อสารและทำงานร่วมกัน ไม่ว่าจะอยู่ในสาขาอาชีพใด การติดต่อสื่อสารและทำงานร่วมกันระหว่างองค์กรนั้นจำเป็นต้องเกิดขึ้น ในอดีตที่ผ่านมาองค์กรต่าง ๆ มักจะประสบปัญหาจากการทำงานร่วมกันโดยใช้แอพพลิเคชั่นต่างๆ ที่ถูกพัฒนามาจากหลากหลายแพลตฟอร์ม หลากหลายระบบปฏิบัติการ หลากหลายภาษา และถึงแม้ว่าองค์กรต่างๆ สามารถที่จะเชื่อมต่อแอพพลิเคชั่นต่างๆ เข้าด้วยกันได้ แต่การทำเช่นนั้นทำให้มีค่าใช้จ่ายที่สูง และมีความสลับซับซ้อนมาก ด้วยเหตุนี้เองจึงมีความต้องการมาตรฐานกลางเพื่อทำให้การติดต่อสื่อสารและทำงานร่วมกันขององค์กรสะดวกและรวดเร็วมากขึ้น มาตรฐานกลางนั่นคือเว็บเซอร์วิส

เว็บเซอร์วิส (Web Service) [1] เป็นระบบซอฟต์แวร์ที่ออกแบบมาเพื่อสนับสนุนการทำงานระหว่างคอมพิวเตอร์กับคอมพิวเตอร์ผ่านระบบเครือข่าย โดยใช้ภาษาเอ็กซ์เอ็มแอล (XML) [2] เป็นภาษาที่ใช้ในการอธิบายการเรียกใช้ซอฟต์แวร์และการติดต่อสื่อสารระหว่างโปรแกรม ตัวอย่างเช่น การบริการในการตรวจสอบราคาหุ้นของตลาดหุ้นหลายๆ ที่และอ่านข่าวจากแหล่งข่าว หลายๆ ที่โดยให้เฉพาะข่าวของบริษัทที่ผู้ขอใช้บริการสนใจ ผู้ให้บริการเว็บเซอร์วิสหนึ่งอาจจะเป็นผู้ขอบริการเว็บเซอร์วิสอื่น ยกตัวอย่างเช่น เว็บเซอร์วิสที่ให้บริการข้อมูลก่อนการซื้อขายหุ้น อาจจะเป็นผู้ขอใช้บริการของเว็บเซอร์วิสที่ให้บริการการให้ข่าว

ความสามารถของเว็บเซอร์วิสที่ทำให้โปรแกรมเชื่อมต่อกับโปรแกรมอื่นได้นั้นเป็นจุดเด่นของเว็บเซอร์วิสที่สามารถจะเชื่อมบริการหลายๆ อันเข้าด้วยกัน ซึ่งความสามารถนี้เกิดจากการใช้ภาษา XML เป็นภาษาในการติดต่อระหว่างผู้พัฒนาเว็บเซอร์วิสและผู้เรียกใช้เว็บเซอร์วิส รูปแบบของข้อมูล XML รูปแบบหนึ่งที่ใช้ในการติดต่อนี้เรียกว่า SOAP (Simple Object Access Protocol) [3] เนื่องจากข้อมูลที่ติดต่ออยู่ในรูปแบบภาษา XML ทำให้โปรแกรมต่าง ๆ สามารถติดต่อกันได้ ถึงแม้ว่าอาจจะถูกพัฒนาและเรียกใช้บนแพลตฟอร์มที่แตกต่างกัน หรือใช้ภาษาที่แตกต่างกันในการพัฒนา ทั้งนี้เนื่องจาก XML เป็นภาษาอักขระ (text) ซึ่งระบบปฏิบัติการทุกระบบสามารถเข้าใจ นอกจากนี้การที่ XML มีแท็ก (tag) และรูปแบบโครงสร้างที่อธิบายข้อมูลด้วยตัวมันเอง ทำให้การเข้าใจและการจัดการข้อความแบบ SOAP นั้นสามารถทำได้โดยโปรแกรมและช่วยทำให้การติดต่อระหว่างผู้ให้บริการและผู้ใช้เว็บเซอร์วิสเป็นไปได้อย่างอัตโนมัติ ในการพัฒนาเซอร์วิสแบบ SOAP นั้นมีมาตรฐานอื่นถูกพัฒนาขึ้นเพื่อสนับสนุนการทำให้เว็บเซอร์วิสติดต่อกันได้อย่างมีประสิทธิภาพมากขึ้น เช่น การใช้เอกสารภาษา WSDL (Web Services Description Language) [4] ซึ่งเป็นภาษา XML ประเภทหนึ่ง โดย WSDL (อ่านว่า วิสเดิล) เป็นภาษาที่ใช้ในการอธิบายการเรียกใช้งานเว็บเซอร์วิสซึ่งเปรียบเสมือนการอ่านคู่มือการใช้งานโปรแกรมนั่นเอง แต่ทว่ามีข้อแตกต่างกันตรงที่ไม่เฉพาะมนุษย์เท่านั้นที่สามารถเข้าใจคู่มือนั่น โปรแกรมที่สามารถอ่านเอกสารภาษา XML เข้าใจสามารถที่จะเข้าใจเอกสาร WSDL ได้เช่นกัน ซึ่งจากคุณสมบัตินี้ช่วยทำให้การเรียกใช้เว็บเซอร์วิสเป็นไปได้อย่างอัตโนมัติ

นอกจากผู้พัฒนาเว็บเซอร์วิสจะพัฒนาเว็บเซอร์วิสแบบ SOAP แล้วผู้พัฒนาเว็บเซอร์วิสยังสามารถพัฒนาเว็บเซอร์วิสแบบ REST ได้ ปัจจุบัน

มีผู้นิยมใช้เว็บเซอร์วิสแบบ REST มากขึ้นดังจะเห็นได้จากสถิติที่บ่งบอกว่า 85% ของผู้ใช้เว็บเซอร์วิสของ Amazon เลือกที่จะใช้อินเตอร์เฟสแบบ REST มากกว่าที่ผู้ใช้เหล่านี้จะเลือกใช้อินเตอร์เฟสแบบ SOAP

ในบทความนี้ในหัวข้อที่ 2 จะได้นำเสนอเว็บเซอร์วิสแบบ REST ส่วนหัวข้อที่ 3 จะมีเนื้อหาในส่วนของแนวทางในการทดสอบฟังก์ชั่นของเว็บเซอร์วิสแบบ REST และบทสรุปอยู่ในหัวข้อที่ 4

2. เว็บเซอร์วิสแบบ REST

REST ย่อมาจาก REpresentational State Transfer เป็นวิธีการที่มองเว็บในฐานะที่เป็นแหล่งข้อมูล (resources) ซึ่งสามารถที่จะจัดการได้โดยวิธีการต่าง ๆ ดังนี้

- GET สำหรับได้การนำเสนอ (representation) ของแหล่งข้อมูล
- DELETE สำหรับลบการนำเสนอ
- PUT สำหรับการสร้างและเปลี่ยนแปลงการนำเสนอ

เว็บเซอร์วิสแบบ REST จะมี URI กำกับเพื่อบอกที่อยู่ของแหล่งข้อมูล ผู้พัฒนาเว็บเซอร์วิสสามารถเลือกใช้ GET/DELETE/POST/PUT ซึ่งแล้วแต่ความเหมาะสมกับแอพพลิเคชั่น

- GET เหมาะสำหรับการดึงข้อมูลออกมาดู
- POST สำหรับการแก้ไขแหล่งข้อมูล

โดยที่ผลของการเรียกเว็บเซอร์วิสจะได้ผลลัพธ์มาอยู่ในรูปแบบของเอกสาร XML
ข้อดีของ REST คือผู้พัฒนาเว็บเซอร์วิสสามารถสร้างและเรียกใช้เว็บเซอร์วิสได้ง่าย เพราะใช้แค่ XML และ HTTP ข้อด้อยของ REST คือการติดต่อขอข้อมูลจะเป็นได้แบบประสานเวลา (synchronous) เท่านั้น และการจัดการความปลอดภัยของข้อมูลเพิ่มเติมจากในส่วนของเว็บเซิร์ฟเวอร์ และการจัดการข้อมูลที่มีความสลับซับซ้อนได้จากการเขียนโค๊ดขึ้นมาเอง

<u>ตัวอย่างการทำงานของเว็บเซอร์วิสแบบ REST โดยใช้เมธอด GET</u>

เว็บเซอร์วิสแบบ REST โดยใช้เมธอด GET จะมีการรับพารามิเตอร์จาก URL ซึ่งจะมีทั้งโอเปอเรชั่นของเว็บเซอร์วิส และค่าต่างๆ ที่ส่งผ่านไปให้โอเปอเรชั่นนั้นๆ จากนั้นเว็บเซอร์วิสจะส่ง REST กลับมาเป็นไฟล์ XML

ตัวอย่างรูปแบบการส่ง REST Request

http://localhost/rest/Calc.php?Operation=add&param1=1&param2=2

Operation=add จะเป็นโอเปอเรชั่นของเว็บเซอร์วิสที่เราสร้างขึ้น ในที่นี้มี 2 ตัว คือ add และ subtract ในตัวอย่างเป็นการเรียกใช้ Opearation=add param1=1&param2=2 เป็นพารามิเตอร์ที่เราต้องใส่เข้าไป ซึ่งจากตัวอย่างจะพบว่าต้องมีพารามิเตอร์ 2 ตัว ซึ่งชื่อว่า param1 และ param2

ผู้ใช้โปรแกรมพิมพ์ request ตรงช่องใส่ URL ของบราวเซอร์เป็น

http://localhost/rest/Calc.php?Operation=add&param1=1&param2=2

หลังจากนั้นผู้ใช้จะได้ผลลัพธ์ออกมาเป็นไฟล์ XML ซึ่งมีเนื้อหาดังนี้

```
<?xml version="1.0"?>
<calculator>
                <operation name="add">

        <result>3</result>
                </operation>
</calculator>
```

3. แนวทางในการทดสอบฟังก์ชั่นของเว็บเซอร์วิสแบบ REST

3.1 เงื่อนไขในการทดสอบฟังก์ชันการทำงานทั่วไป

1. ตรวจสอบความถูกต้องในการเรียกใช้ในแต่ละฟังก์ชัน
    - ทดลองส่งค่าตัวแปรที่ถูกต้องให้แต่ละฟังก์ชัน
    - ทดลองเรียกใช้งานแต่ละฟังก์ชัน เพื่อดูผลว่า สามารถเรียกใช้ได้หรือไม่
2. ตรวจสอบความครบถ้วนของข้อมูลที่ได้จากการเรียกใช้เว็บเซอร์วิสในแต่ละฟังก์ชัน
    - ทดสอบเหมือนข้อ 1
    - ตรวจสอบผลลัพธ์ที่ได้จากการเรียกใช้แต่ละฟังก์ชันเทียบกับโครงสร้างข้อมูลในเอกสาร Web Services REST API GUIDE
3. ตรวจสอบการแสดงข้อผิดพลาดเมื่อส่งค่าตัวแปรที่ไม่ถูกต้องในแต่ละฟังก์ชัน
    - ทดลองใส่ค่าข้อมูลที่ไม่ถูกต้องลงในแต่ละฟังก์ชัน
        - สุ่มการใส่ข้อมูลที่ไม่ถูกต้อง แล้วรายงานผล
    - ตรวจสอบข้อผิดพลาดที่ได้กับเอกสาร Web Services REST API GUIDE
4. ตรวจสอบการแสดงข้อผิดพลาดเมื่อไม่ส่งค่าตัวแปรในแต่ละฟังก์ชัน
    - ไม่ใส่ค่าอินพุตพารามิเตอร์แล้วทำการเรียกใช้แต่ละฟังก์ชัน
    - ตรวจสอบข้อความแสดงข้อผิดพลาดที่คาดว่าจะได้รับ
5. ตรวจสอบการแสดงข้อผิดพลาดเมื่อส่งคีย์ Authentication ไม่ถูกต้องในแต่ละฟังก์ชัน
    - ทดลองสุ่ม Key Authentication แล้วทำการเรียกใช้แต่ละฟังก์ชัน
    - ทดสองแก้ไข Key Authentication เดิม แล้วทำการเรียกใช้แต่ละฟังก์ชัน
    - ตรวจสอบข้อความแสดงข้อผิดพลาดที่คาดว่าจะได้รับ
6. ตรวจสอบการแสดงข้อผิดพลาดเมื่อไม่มีการส่งคีย์ Authentication ในแต่ละฟังก์ชัน
    - ทดลองทำการเรียกใช้แต่ละฟังก์ชันโดยไม่ทำการส่ง Key Authentication
    - ตรวจสอบข้อความแสดงข้อผิดพลาดที่คาดว่าจะได้รับ
7. ตรวจสอบการแสดงข้อผิดพลาด เมื่อส่งค่าอินพุตไม่ครบตามจำนวนที่ฟังก์ชันต้องการในแต่ละฟังก์ชัน
    - ทดลองเรียกใช้แต่ละฟังก์ชันโดยส่งอินพุตไม่ครบตามจำนวนฟังก์ชันที่ต้องการ
    - ตรวจสอบข้อความแสดงข้อผิดพลาดที่คาดว่าจะได้รับ
8. ตรวจสอบการแสดงข้อผิดพลาด เมื่อส่งค่าอินพุตมากกว่าที่จำนวนที่ฟังก์ชันต้องการในแต่ละฟังก์ชัน
    - ทดลองเพิ่มพารามิเตอร์ที่ทำการเรียกใช้แต่ละฟังก์ชัน
    - ตรวจสอบข้อความแสดงข้อผิดพลาดที่คาดว่าจะได้รับ

9. ตรวจสอบการแสดงข้อผิดพลาด เมื่อส่งค่าตัวแปรคนละประเภทที่ฟังก์ชันต้องการในแต่ละฟังก์ชัน
   - ทดลองส่งค่าตัวแปรที่ต้องการจากชนิดอื่นเป็น String
   - ทดลองส่งค่าตัวแปรที่ต้องการจากชนิดสลับซับซ้อน (complex type) เป็นชนิดง่าย (simple type)
   - ตรวจสอบข้อความแสดงข้อผิดพลาดที่คาดว่าจะได้รับ
10. ตรวจสอบค่าของข้อมูลเมื่อส่งค่าอินพุตเป็น Null ในแต่ละฟังก์ชัน
   - ทดลองส่งค่าตัวแปรเป็น Null ในแต่ฟังก์ชัน
   - ระบบต้องไม่มีการแสดงผลข้อมูลที่ถูกต้อง
11. ตรวจสอบการแสดงข้อผิดพลาดเมื่อส่งค่าอินพุตเป็น Null ในแต่ละฟังก์ชัน
   - ทดลองเหมือนข้อ 10
   - ตรวจสอบข้อความแสดงข้อผิดพลาดที่คาดว่าจะได้รับ
12. ตรวจสอบความถูกต้องของข้อมูลเมื่อข้อมูลเป็นชนิดข้อมูลที่ซับซ้อนในแต่ละฟังก์ชัน
   - ทดลองส่งข้อมูลในรูปแบบซับซ้อน
   - ตรวจสอบข้อมูลที่ได้รับ
13. ตรวจสอบการแสดงข้อผิดพลาดเมื่อผู้ใช้ขอบริการข้อมูลที่ไม่มีให้บริการในแต่ละฟังก์ชัน
   - ทดลองส่งข้อมูลที่ระบบไม่มีให้บริการ
   - ตรวจสอบข้อความแสดงข้อผิดพลาดที่คาดว่าจะได้รับ
14. ตรวจสอบรูปแบบของชนิดข้อมูล XML ว่าถูกต้องตามโครงสร้างของเอกสารหรือไม่ (Valid Document)
   - ทดลองเหมือนข้อ 1
   - ตรวจสอบผลลัพธ์ เทียบกับชนิดข้อมูลในเอกสาร Web Services REST API GUIDE
15. ตรวจสอบข้อมูลที่ได้จากเว็บเซอร์วิสว่ามีรูปแบบตรงตามกฎภาษา XML หรือไม่ (Well-formed XML Document)
   - ทดลองเหมือนข้อ 1
   - ตรวจสอบผลลัพธ์ที่ได้ ถ้าสามารถแสดงผลผ่านทางบราวเซอร์ได้ แสดงว่าเป็น Well-formed XML Document

3.2 แนวทางในการทดสอบ

1. เขียนโปรแกรมเรียกใช้เว็บเซอร์วิสโดยใช้ภาษา Java, PHP และ .NET
2. ทำการทดลองเรียกใช้ฟังก์ชัน ตามเงื่อนไขข้างต้น
3. ตรวจสอบผลลัพธ์ และรายงานผลที่ได้

3.3 สภาพแวดล้อมการทดสอบ

1. เครื่องเซิร์ฟเวอร์ต้องเปิดอยู่
2. โปรแกรมเว็บเซิรฟ์เวอร์ ที่เป็น IIS หรือ Apache HTTP จะต้องเปิดให้บริการอยู่
3. เว็บเซอร์วิสแบบ REST จะต้องถูกติดตั้งเรียบร้อยแล้วบนเซิร์ฟเวอร์
4. เซิร์ฟเวอร์ที่เป็นฐานข้อมูล จะต้องเปิดให้บริการอยู่
5. เครื่องผู้ทดสอบจะต้องมีโปรแกรมเว็บบราวเซอร์ หรือ สามารถเชื่อมต่ออินเตอร์เน็ตได้

3.4 ตัวอย่างการทดสอบฟังก์ชัน

ตัวอย่างการทดสอบนี้ได้ทดสอบจริงที่เว็บ http://service.a2z-protravel.com/ws/rest/DestinationList.php

หมายเลขกรณีทดสอบ	SPC01_01	**ชื่อ**	ตรวจสอบความถูกต้องในการเรียกใช้ฟังก์ชัน DestinationList
ทดสอบความต้องการ	เพื่อทดสอบความถูกต้องของข้อมูลที่ได้รับเมื่อมีการเรียกใช้ฟังก์ชัน DestinationList		
วัตถุประสงค์	เพื่อทดสอบความถูกต้องของข้อมูลที่ได้รับเมื่อมีการเรียกใช้ฟังก์ชัน ว่ามีความถูกต้องตามที่ระบุไว้ในคู่มือจริงหรือไม่		
คำอธิบาย	1. เขียนโปรแกรมเรียกใช้งานเว็บเซอร์วิส 2. เรียกใช้ฟังก์ชัน DestinationList 3. กรอกข้อมูลที่ถูกต้องและมีในฐานข้อมูล โดยให้ค่าของตัวแปรดังนี้ - Operation มีค่า getDestinationList - Key มีค่า dGVzdGVyQGEyei1wcm90cmF2ZWwuY29tJnRlc3Rlcg== - provinceName มีค่า Bangkok		
เงื่อนไขในการทดสอบ	o ระบบต้องมีค่าข้อมูลที่ทำการค้นหา		
ผลลัพธ์ที่คาดว่าจะได้รับ	o ระบบคืนค่าข้อมูลที่ถูกต้องได้		
ผลลัพธ์ที่เกิดขึ้นจริง	o ระบบคืนค่าข้อมูลที่ถูกต้อง ตามรูปแบบในเอกสาร A2Z Web Services REST API GUIDE		
กรณีผิดพลาด	-		
ผลการทดสอบ	☑ ผ่าน	☐ ไม่ผ่าน	
หมายเหตุ	-		
ผู้ทดสอบ	1. เสกสิทธิ์ สุวรรณ 2. ภาณุวัฒน์ คามวัลย์ 3. ชัยวัฒน์ บุตรไชย		

3.5 ตัวอย่างการสรุปผลการทดสอบ แยกตามกรณีทดสอบ

1. ***ตรวจสอบความถูกต้องของข้อมูลโดยเรียกใช้ในแต่ละฟังก์ชัน***

การทดสอบกรณีนี้ อ้างอิงตามเอกสาร Web Services REST API GUIDE ในส่วนของตาราง Response ของทุกฟังก์ชัน ดังนั้น หากฟังก์ชันใดมีจำนวนอิลีเมนต์ขาดหรือเกิน ถือว่าไม่ผ่านในหัวข้อนี้

ฟังก์ชันที่ไม่ผ่านการทดสอบ ได้แก่ HotelInformation และ RoomList

2. ***ตรวจสอบความครบถ้วนของข้อมูลที่ได้จากการเรียกใช้เว็บเซอร์วิสในแต่ละฟังก์ชัน***

การทดสอบในกรณีนี้ใช้การอ้างอิงจากผลลัพธ์ที่ได้ ซึ่งถ้าไม่มีข้อความแสดงข้อผิดพลาดถือว่าเป็นผลลัพธ์ที่ถูกต้อง และครบถ้วน

ผลการทดสอบพบว่า ผ่านทุกฟังก์ชัน

3. ***ตรวจสอบการแสดงข้อผิดพลาดเมื่อส่งค่าข้อมูลที่ไม่ถูกต้องในแต่ละฟังก์ชัน***

การทดสอบในกรณีนี้ ทดสอบโดยการส่งค่าอินพุตที่ระบบไม่มีผลลัพธ์ เพื่อให้ระบบแสดงข้อความแสดงข้อผิดพลาดกลับมา แล้วตรวจสอบข้อความที่แสดงข้อผิดพลาดกับเอกสารว่ามีความถูกต้องตามเอกสารหรือไม่ พบว่าบางฟังก์ชันมีการส่งข้อความที่แสดงข้อผิดพลาดได้ถูกต้องตามเอกสาร แต่ยังมีบางฟังก์ชันที่ไม่แสดงตามที่เอกสารได้ระบุไว้

ฟังก์ชันที่ไม่ได้แสดงข้อความแสดงข้อผิดพลาดตามเอกสารนั้น พบว่า เมื่อใส่อินพุตในตัวแปรบางตัว ฟังก์ชันนั้นยังสามารถทำงานได้ปกติ ซึ่งได้แก่ ฟังก์ชัน Reservation, Cancellation และในฟังก์ชัน DestinationList ที่ไม่ผ่านเนื่องจากมีข้อผิดพลาดที่ไม่ตรงตามเอกสาร

ฟังก์ชันที่ไม่ผ่านการทดสอบ ได้แก่ DestinationList, Reservation และ Cancellation

4. ***ตรวจสอบการแสดงข้อผิดพลาดเมื่อไม่ส่งค่าข้อมูลในแต่ละฟังก์ชัน***

การทดสอบกรณีนี้ ทดสอบโดยไม่ทำการส่งค่าเข้าไปในฟังก์ชัน และคาดหวังให้ระบบแสดงข้อความแสดงข้อผิดพลาดออกมา

ผลการทดสอบพบว่า ผ่านทุกฟังก์ชัน

5. ***ตรวจสอบการแสดงข้อผิดพลาดเมื่อส่งคีย์ Authentication ไม่ถูกต้องในแต่ละฟังก์ชัน***

การทดสอบกรณีนี้ ทดสอบโดยการส่งค่า Key Authentication ที่ผิดไปให้ระบบ และคาดหวังให้ระบบแจ้งข้อความแสดงข้อผิดพลาดออกมา ซึ่งพบว่าทุกฟังก์ชันไม่สามารถผ่านในกรณีนี้ไปได้

ผลการทดสอบพบว่า ไม่ผ่านทุกฟังก์ชัน

6. ***ตรวจสอบการแสดงข้อผิดพลาดเมื่อไม่มีการส่งคีย์ Authentication ในแต่ละฟังก์ชัน***

การทดสอบกรณีนี้ ทดสอบโดยที่ไม่ทำการส่งค่า Key Authentication เข้าไปในฟังก์ชัน แต่ยังคงใส่ข้อมูลต่างๆ ครบทุกตัว

ผลการทดสอบพบว่า ผ่านทุกฟังก์ชัน

7. ***ตรวจสอบการแสดงข้อผิดพลาด เมื่อส่งค่าอินพุตไม่ครบตามจำนวนที่ฟังก์ชันต้องการในแต่ละฟังก์ชัน***

การทดสอบกรณีนี้ ทำการทดสอบโดยส่งอินพุตไม่ครบ และอ้างอิงตามเอกสาร โดยดูว่าเป็นตัวแปรที่จำเป็นต้องมี (required variables) หรือเป็นตัวแปรที่มีหรือไม่มีก็ได้ (optional variables) ฟังก์ชันที่ไม่ผ่านส่วนใหญ่พบว่า ในเอกสารระบุว่าตัวแปรที่จำเป็นต้องมีแต่เมื่อทดลองไม่ส่งค่าตัวแปรที่จำเป็นต้องมีไปให้กับเว็บเซิร์ฟเวอร์ กลับพบว่าฟังก์ชั่นนั้นสามารถทำงานได้ตามปกติ

ฟังก์ชันที่ไม่ผ่านการทดสอบ ได้แก่ HotelList, CheckAvailability, RoomList, Reservation และ Cancellation

8. ***ตรวจสอบการแสดงข้อผิดพลาด เมื่อส่งค่าอินพุตมากกว่าที่จำนวนที่ฟังก์ชันต้องการในแต่ละฟังก์ชัน***

การทดสอบกรณีนี้ เพื่อดูผลว่า หากมีการส่งค่าตัวแปรเกินกว่าที่ฟังก์ชันต้องการ หรือส่งค่าตัวแปรที่ซ้ำกันมาให้ ระบบควรจะแสดงข้อความแสดงข้อผิดพลาดกลับมา

ผลการทดสอบพบว่า ไม่ผ่านทุกฟังก์ชัน

9. ***ตรวจสอบการแสดงข้อผิดพลาด เมื่อส่งค่าข้อมูลคนละประเภทที่ฟังก์ชันต้องการในแต่ละฟังก์ชัน***

การทดสอบในกรณีนี้ ทำการทดลองใส่ค่าตัวแปรคนละประเภทที่ฟังก์ชันต้องการ เช่น จากประเภท num ส่งเป็น String, ประเภท Data ส่งเป็น String ประเภท complex type ส่งเป็น simple type และดู error message ที่ได้พบว่า ฟังก์ชันที่ไม่ผ่านเมื่อทำการส่งค่าอินพุตแต่ละประเภทไปให้ฟังก์ชันนั้น ฟังก์ชันนั้นๆ ยังคงทำงานปกติ (รายละเอียดดูในการทดสอบระบบ)

ฟังก์ชันที่ไม่ผ่านการทดสอบ ได้แก่ HotelList, Reservation และ Cancellation

10. ***ตรวจสอบค่าของข้อมูลเมื่อส่งค่าอินพุตเป็น Null ในแต่ละฟังก์ชัน***

การทดสอบในกรณีนี้ ทำการทดลองใส่ค่าที่ฟังก์ชันต้องการให้เป็นค่าเป็น “Null” ฟังก์ชันไม่ควรที่จะทำงานได้ตามปกติ และควรส่ง error message กลับมา แต่การทดสอบพบว่า มีบางฟังก์ชันที่ยังสามารถทำงานได้ตามปกติ (รายละเอียดดูในการทดสอบระบบ)

ฟังก์ชันที่ไม่ผ่านการทดสอบ ได้แก่ CheckAvailability, RoomList, Reservation และ Cancellation

11. ***ตรวจสอบการแสดงข้อผิดพลาดเมื่อส่งค่าอินพุตเป็น Null ในแต่ละฟังก์ชัน***

การทดสอบในกรณีนี้ เพื่อดูข้อความที่แสดงข้อผิดพลาดที่ได้รับ ทำการทดสอบเหมือนกับกรณีที่ 10 แต่เป็นการดูผลลัพธ์ต่างกัน

ฟังก์ชันที่ไม่ผ่านการทดสอบ ได้แก่ CheckAvailability, RoomList, Reservation และ Cancellation

12. ตรวจสอบความถูกต้องของข้อมูลเมื่อข้อมูลเป็นชนิดข้อมูลที่ซับซ้อนในแต่ละฟังก์ชัน

การทดสอบกรณีนี้ เป็นการผลลัพธ์ที่ได้ในแต่ละฟังก์ชันว่าเป็นข้อมูลชนิดซับซ้อนหรือไม่

ผลการทดสอบพบว่า ผ่านทุกฟังก์ชัน

13. ตรวจสอบการแสดงข้อผิดพลาดเมื่อผู้ใช้ขอบริการข้อมูลที่ไม่มีให้บริการในแต่ละฟังก์ชัน

การทดสอบกรณีนี้ ทำการทดลองใส่ข้อมูลที่ฟังก์ชันไม่มีผลลัพธ์ให้ และจะต้องทำการส่งข้อความแสดงข้อผิดพลาดกลับมา แต่ยังมีบางฟังก์ชันที่ยังไม่ผ่านในกรณีนี้ (รายละเอียดดูในการทดสอบระบบ)

ฟังก์ชันที่ไม่ผ่านการทดสอบ ได้แก่ Cancellation

13. ตรวจสอบรูปแบบของชนิดข้อมูล XML ว่าถูกต้องตามโครงสร้างของเอกสารหรือไม่ (Valid Document)

การทดสอบกรณีนี้ อ้างอิงจากเอกสาร เนื่องจากไม่มี XML Schema ให้เปรียบเทียบ จึงจำเป็นต้องอาศัยการอ้างอิงจากตาราง response ว่าชนิดข้อมูลที่ได้กลับมา มีความถูกต้องหรือไม่

จากการทดสอบในกรณีที่ 1 ทำให้พบว่า มีบางฟังก์ชันที่มีจำนวนแท็กของอิลีเมนต์เกิน ซึ่งไม่ถูกต้องตามโครงสร้างเอกสาร (ตามการอ้างอิง)

ฟังก์ชันที่ไม่ผ่านการทดสอบ ได้แก่ HotelInformation และ CheckAvailability

14. ตรวจสอบข้อมูลที่ได้จากเว็บเซอร์วิสว่ามีรูปแบบตรงตามกฎภาษา XML หรือไม่ (Well-formed XML Document)

การทดสอบกรณีนี้ สามารถตรวจสอบได้โดยการใช้ เว็บบราวเซอร์ ได้ เพราะว่าหากเป็น XML ที่ Well-formed แล้ว จะสามารถแสดงผลออกทางเว็บบราวเซอร์ได้

ผลการทดสอบพบว่า ผ่านทุกฟังก์ชัน

4. สรุป

ในบทความนี้ได้นำเสนอแนวทางในการทดสอบความสมบูรณ์และความถูกต้องของฟังก์ชั่นของเว็บเซอร์วิสแบบ REST โดยที่ได้นำเสนอตัวอย่างของแบบทดสอบและตัวอย่างการสรุปผลการทดสอบ การทดสอบเว็บเซอร์วิสนั้นจำเป็นต้องมีเพื่อส่งเสริมให้ผู้พัฒนาเว็บเซอร์วิสสามารถพัฒนาเว็บเซอร์วิสได้สมบูรณ์และถูกต้อง ซึ่งจะมีผลทำให้ช่วยสนับสนุนให้เกิดการพัฒนาและการใช้งานเว็บเซอร์วิสอย่างแพร่หลายในประเทศไทย

อย่างไรก็ตาม การทดสอบเว็บเซอร์วิสนั้นไม่ควรจะทดสอบเพียงแค่ฟังก์ชั่นการใช้งานของเว็บเซอร์วิสเท่านั้น อย่างเช่น ควรจะทดสอบความเร็ว (response time) และความปลอดภัยของข้อมูล (security)

5. เอกสารอ้างอิง

[1] W3C, “Web Services Architecture”, http://www.w3.org/TR/ws--arch/

[2] W3C, “Extensible Markup Language (XML)”, http://www.w3.org/XML/

[3] W3C, “SOAP Version 1.2 Part 0: Primer”, http://www.w3.org/TR/2003/REC-soap12-part0-20030624/

[4] W3C, “Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language”, http://www.w3.org/TR/2004/WD-wsdl20-20040326/

[5] Roy T. Fielding , “Architectural Styles and the Design of Network-based Software Architectures ”, http://portal.acm.org/citation.cfm?id=932295