**SOAP ย่อมาจาก Simple Object Access Protocol**  
ถูกกำหนดขึ้นมาตั้งแต่ปี 1998  
เป้าหมายเพื่อใช้ในตลาด enterprise โดยเฉพาะ

ปล. developer หลาย ๆ คนอาจจะไม่รู้จัก SOAP ก็เป็นได้ !!

**REST ย่อมาจาก Representational State Transfer**  
ถูกกำหนดขึ้นมาตั้งแต่ปี 2000  
เป้าหมายเพื่อเป็นรูปแบบหนึ่งในการออกแบบ open web technology

แนวคิดพื้นฐาน

**SOAP นั้นต้องสร้างให้ application logic ออกมาเป็น service**  
ทำให้การตั้งชื่อเป็นดังนี้

* getUserInformation
* payInvoice

โดย SOAP เป้าหมายเพื่อเป็น protocol ใหม่ของการติดต่อสื่อสาร

**ส่วน REST นั้นต้องการทำให้ข้อมูลอยู่ในรูปแบบของ resource**  
ส่วนการกระทำต่าง ๆ เป็นไปตาม HTTP Verb หรือ HTTP Method (GET, POST, PUT, DELETE)  
และทำงานแบบ Stateless  
ทำให้การตั้งชื่อเป็นดังนี้

* userinformation
* invoice

ข้อดีและข้อเสียของ SOAP

ข้อดี

* สามารถทำงานอยู่บน protocol ใด ๆ ก็ได้
* อธิบาย service ด้วย WDSL (Web Service Description Language)
* มีความน่าเชื่อถือ เมื่อเกิดปัญหาสามารถทำการ retry ได้
* สนับสนุนเรื่อง security อยู่แล้ว ทั้ง authentication, authorization และ การเข้ารหัสข้อมูล

ข้อเสีย

* ยากต่อการพัฒนา ทำให้ไม่เป็นที่นิยมสำหรับระบบ web และ mobile
* สนับสนุนรูปแบบข้อมูล XML เพียงอย่างเดียว
* เนื่องจากมันเป็น standard ทำให้มีข้อจำกัดเยอะ
* เนื่องจากโครงสร้างมันมีหลายส่วนทำให้มี overhead สูง หรือ ต้องใช้ bandwidth สูงกว่า REST

ข้อดีและข้อเสียของ REST

ข้อดี

* ทำการอยู่บน HTTP และทำตามมาตรฐานของ HTTP จึงทำให้พัฒนาได้ง่าย
* สนับสนุนรูปแบบข้อมูลมากมาย เช่น XML, JSON, Plain Text และอื่น ๆ อีกมากมาย
* รองรับการขยายระบบได้ง่าย
* มีประสิทธิภาพการทำงานที่ดี
* รองรับเรื่อง caching ข้อมูล

ข้อเสีย

* ทำงานได้เฉพาะ HTTP protocol เท่านั้น
* ไม่มีเรื่องของ security และ reliability มาให้ในตัว ดังนั้นต้องทำเอง
* รูปแบบข้อมูลที่ส่งไปมาระหว่าง client-server ไม่มีข้อจำกัดอะไรเลย

คำถามที่น่าสนใจคือ จะใช้ SOAP และ REST เมื่อใดบ้าง ?

**ใช้ SOAP เมื่อ**  
ต้องการจัดการ transaction เมื่อต้องทำงานกับหลาย ๆ ระบบ  
ต้องการความเข้มงวดในการเชื่อมต่อระหว่าง client/server  
ตัวอย่างเช่น Finacial service และ Telecommunication service  
ดังนั้นจึงไม่แปลกว่า ทำไมในองค์กรใหญ่ ๆ เหล่านี้ใช้ SOAP กันมาก

**ใช้ REST เมื่อ**  
ต้องการลดขนาดของข้อมูล และ จำนวน bandwidth ที่ใช้งาน  
ต้องการเมื่อทำงานอยู่บนระบบ web และ mobile  
ตัวอย่างเช่น Social media service, Web Chat service  
ดังนั้นจึงไม่แปลกว่า ทำไมระบบ web และ api ต่าง ๆ ผ่าน web จึงเป็น REST

**บางคนไปหาข้อมูลสำหรับ Open API น่าจะข้อมูลดังนี้**  
ซึ่งบอกว่า REST มันได้รับความนิยมกว่า SOAP มากมาย

สุดท้ายแล้ว ก่อนจะเลือกใช้งานอะไร

ต้องมีความรู้พื้นฐานก่อน  
ต้องมีความเข้าใจก่อน  
ต้องเข้าใจความต้องการของระบบก่อน  
จึงจะเลือกได้อย่างถูกต้องและเหมาะสม

**SOAP คืออะไร?**

SOAP ย่อมาจาก Simple Object Access Protocol ถูกกำหนดขึ้นมาตั้งแต่ปี 1998 เป้าหมายเพื่อใช้ในตลาด enterprise โดยเฉพาะ SOAP นั้นต้องสร้างให้ application logic ออกมาเป็น service โดยเป้าหมายเพื่อเป็น protocol ใหม่ของการติดต่อสื่อสาร

**ข้อดี**

- สามารถทำงานอยู่บน protocol ใด ๆ ก็ได้

- อธิบาย service ด้วย WDSL (Web Service Description Language)

- มีความน่าเชื่อถือ เมื่อเกิดปัญหาสามารถทำการ retry ได้

- สนับสนุนเรื่อง security อยู่แล้ว ทั้ง authentication, authorization และ การเข้ารหัสข้อมูล

**ข้อเสีย**

- ยากต่อการพัฒนา ทำให้ไม่เป็นที่นิยมสำหรับระบบ web และ mobile

- สนับสนุนรูปแบบข้อมูล XML เพียงอย่างเดียว

- เนื่องจากมันเป็น standard ทำให้มีข้อจำกัดเยอะ

- เนื่องจากโครงสร้างมันมีหลายส่วนทำให้มี overhead สูง หรือ ต้องใช้ bandwidth สูงกว่า REST

**เมื่อไรถึงควรใช้ SOAP**

- เมื่อต้องการจัดการ transaction เมื่อต้องทำงานกับหลาย ๆ ระบบ

- เมื่อต้องการความเข้มงวดในการเชื่อมต่อระหว่าง client/server

- เมื่อตัวอย่างเช่น Finacial service และ Telecommunication service

ดังนั้นจึงไม่แปลกว่า ทำไมในองค์กรใหญ่ ๆ เหล่านี้ใช้ SOAP กันมาก

**REST คืออะไร?**

REST ย่อมาจาก Representational State Transfer ถูกกำหนดขึ้นมาตั้งแต่ปี 2000 เป้าหมายเพื่อเป็นรูปแบบหนึ่งในการออกแบบ open web technology โดยต้องการทำให้ข้อมูลอยู่ในรูปแบบของ resource ส่วนการกระทำต่าง ๆ เป็นไปตาม HTTP Verb หรือ HTTP Method (GET, POST, PUT, DELETE) และทำงานแบบ Stateless

**ข้อดี**

- ทำการอยู่บน HTTP และทำตามมาตรฐานของ HTTP จึงทำให้พัฒนาได้ง่าย

- สนับสนุนรูปแบบข้อมูลมากมาย เช่น XML, JSON, Plain Text และอื่น ๆ อีกมากมาย

- รองรับการขยายระบบได้ง่าย

- มีประสิทธิภาพการทำงานที่ดี

- รองรับเรื่อง caching ข้อมูล

**ข้อเสีย**

- ทำงานได้เฉพาะ HTTP protocol เท่านั้น

- ไม่มีเรื่องของ security และ reliability มาให้ในตัว ดังนั้นต้องทำเอง

- รูปแบบข้อมูลที่ส่งไปมาระหว่าง client-server ไม่มีข้อจำกัดอะไรเลย

**เมื่อไรถึงควรใช้ REST**

- เมื่อต้องการลดขนาดของข้อมูล และ จำนวน bandwidth ที่ใช้งาน

- เมื่อต้องการเมื่อทำงานอยู่บนระบบ web และ mobile ตัวอย่างเช่น Social media service, Web Chat service

ดังนั้นจึงไม่แปลกว่า ทำไมระบบ web และ api ต่าง ๆ ผ่าน web จึงเป็น REST