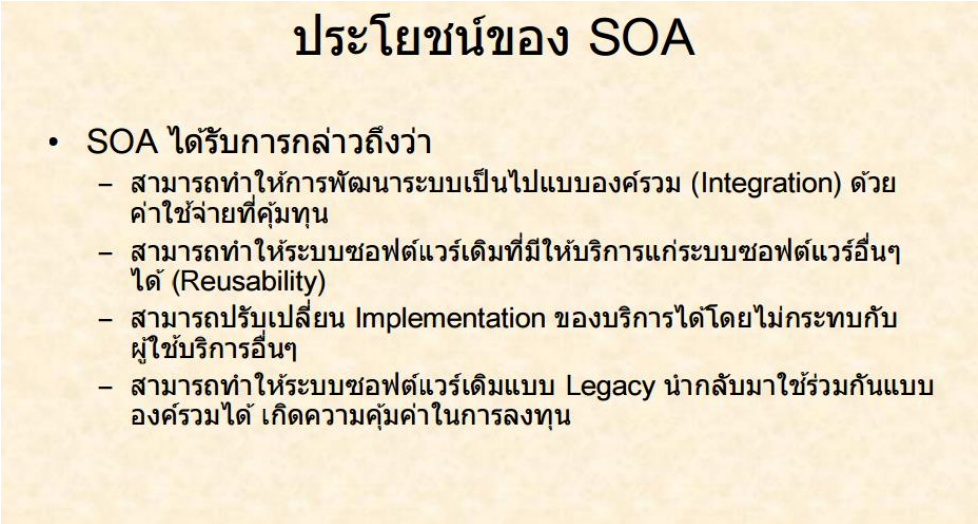
SERVICE

1.SOA คืออะไร

ANS. เป็นสถาปัตยกรรมเชิงบริการ ซึ่งเป็นแนวคิดในการออกแบบระบบ หรือซอฟต์แวร์ ขององค์กรในลักษณะเชิงบริการ โดย services เหล่านี้เป็นอิสระต่อกัน มีฟังก์ชันการทำงานที่มีขอบเขตชัดเจน มุ่งเน้นให้มีการนำสารสนเทศที่มีอยู่เดิมมาใช้ให้เกิดประโยชน์สูงสุด ด้วยการทำให้ทรัพยากรที่มีความหลากหลายและซับซ้อนสามารถทำงานร่วมกันได้ โดยบริการต่างๆของซอฟต์แวร์ที่ถูกพัฒนาขึ้นนั้นสามารถสื่อสารกันด้วยมาตรฐานที่เป็นที่ยอมรับทั่วไป

SOA? แบ่งเป็น 2 คำ Service-Oriented และ Architecture
คำแรก Service-Oriented เป็น Software ที่ไม่ใช่ซอฟต์แวร์ แพ็คเกจ แต่เป็นซอฟต์แวร์ตัวเล็ก ทำงานเฉพาะด้าน ขึ้นอยู่กับว่าจะแบ่งเป็นบริการอะไรบ้าง
คำที่สอง Architecture คือการออกแบบ โดยจะมององค์กรโดยรวมว่าต้องการบริการอะไรบ้าง ก็จะแบ่งบริการนั้นๆออกเป็นส่วนย่อยๆ
ทั้งนี้ หลายคนมองว่า SOA? คือ web service แต่จริงๆแล้วไม่ใช่เพราะ web service เป็นแค่เครื่องมือในการใช้งาน
ดังนั้น SOA จึงไม่ใช่สินค้า หาซื้อไม่ได้ แต่มันคือแนวคิดที่ต้องสร้างเองในองค์กร



2.ความแตกต่างระหว่าง REST กับ SOAP

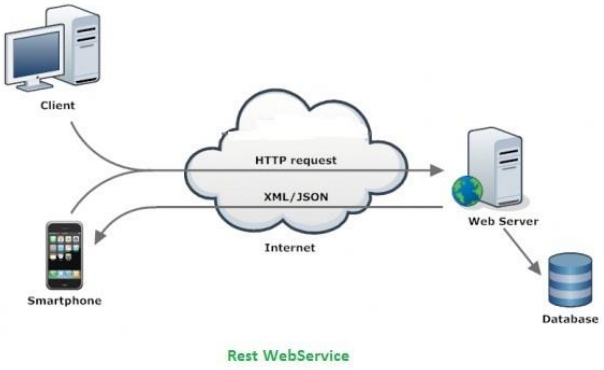
ANS. SOAP และ REST เกี่ยวข้องกับ Web Service โดย SOAP ย่อมาจาก "Simple Object Access Protocol" และ REST "Representational State Transfer"

ในการพัฒนาเว็บเซอร์วิสนั้นเราสามารถเลือกที่จะพัฒนาแบบ SOAP หรือแบบ REST ก็ได้ ถ้าเราพัฒนา SOAP Web services เราจะต้องมีการส่งข้อความ XML (เอกซ์เอ็มแอล) ตามรูปแบบที่กำหนดไว้โดยโปรโตคอล SOAP อีกทั้งต้องมีเอกสารอธิบายการเรียกใช้เว็บเซอร์วิสประกอบ ซึ่งเอกสารที่อธิบายนี้จะเขียนโดยใช้ภาษา WSDL (วิสเดิล) ในแง่ของผู้เรียกใช้ จะต้องมีการเข้าใจเอกสารที่อธิบายการเรียกใช้ SOAP Web services หรือมีเครื่องมือที่จะเข้าใจและเรียกใช้ได้อย่างถูกต้อง

ในขณะที่ REST Web service จะเป็นรูปแบบของซอฟต์แวร์ที่มองว่าข้อมูลต่าง ๆ เป็น Resource ซึ่งคนสามารถเรียกใช้ได้ผ่านทางโปรโตคอล HTTP และข้อมูลที่ส่งกลับมาให้ผู้ใช้เป็นข้อมูลรูปแบบ XML ใด ๆ ก็ได้ ในแง่ของผู้เรียกใช้ REST Web service ก็ขอเพียงแค่ให้ทราบ URL ของ REST Web service และการอ่านข้อมูล XML ก็จะดึงข้อมูลที่ตนเองต้องการได้

3. วาดรูป Architecture ที่เรารู้จัก1อัน และทำไมต้องเป็นอันนี้

ANS.



Rest WebService

Cross platform

Open standard

W3C Standards

Web Technology

HTTP Protocol

Port 80

4.ทำไมต้องdesign architecture แล้วคนทำarchitectureต้องสนใจประเด็นใดบ้าง

ANS.

- Reusability ความง่ายในการเรียนรู้ที่จะใช้ระบบ
- Scalability ความสามารถในการรองรับการเพิ่มขยายได้ในอนาคต
- Maintainability ความง่ายต่อการดัดแปลงแก้ไขระบบ เพื่อปรับเปลี่ยนแปลงหรือเพิ่มความสามารถ ปรับปรุงสมรรถนะหรือแก้ไขข้อบกพร่อง
- Flexibility ขอบเขตที่สามารถดัดแปลงแก้ไขระบบสำหรับการใช้งานหรือสภาวะแวดล้อมอื่นนอกเหนือจากที่ได้รับการออกแบบมาโดยเฉพาะ

5.ทำไมต้องทำ Software Architecture มันต่างจาก Detail design ยังไง

ANS. สถาปัตยกรรมซอฟต์แวร์ คือ การอธิบายภาพรวมของระบบซึ่งมีโครงสร้างที่เชื่อมโยงองค์ประกอบสำคัญ ที่มีผลต่อภาพรวมของทั้งระบบ ทั้งในด้านฟังก์ชั่นและคุณภาพ ซึ่งแสดงให้เห็นถึงผลกระทบ, ผลสะท้อน, คุณสมบัติ และคุณลักษณะ ของการเชื่อมกันระหว่างองค์ประกอบเหล่านั้น โดยมุ่งเน้นจัดการและอธิบายในจุดสำคัญที่มีผลต่อความสนใจด้านธุรกิจและด้านเทคนิค และมุ่งเน้นวางกรอบแนวคิดพื้นฐานกำกับให้ผู้รับผิดชอบในส่วนต่างๆ ยึดปฏิบัติและต่อยอดเพื่อให้ระบบมีเอกภาพ ตอบโจทย์ภาพรวมเดียวกัน

Detailed Design

หลังจากที่ได้โครงสร้างพื้นฐาน (System architecture) แล้ว ถัดมาก็คือการออกแบบระบบในระดับที่ลึกลงไปอีก ผลลัพธ์ที่ได้จากกระบวนการนี้จะเป็นแนวทางอย่างดีสำหรับ Developer ในการเขียนโค๊ด

6. loosely-coupled

Ans. Loosely Coupled หมายถึง

- Service Implementation แยกออกจาก Service Interface
- ระบบงานสองระบบที่เป็นอิสระต่อกัน แต่มีความสัมพันธ์ต่อกันอย่างหลวมๆ

Why we need loosely-coupled?

Requirement is always changed

**More loosely-coupled, more maintainable

7. Components as services

ANS.

8. Services integration

ANS.

9. Web Service Concept

ANS. Web Services คืออะไร

Application หรือ program ที่ทำงานอย่างใดอย่างหนึ่ง ในลักษณะให้บริการ โดยจะถูกเรียกใช้งานจาก application อื่นๆ ในรูปแบบ RPC (Remote Procedure Call) ซึ่งการให้บริการจะมีเอกสารที่อธิบายคุณสมบัติของบริการกำกับไว้ โดยภาษาที่ถูกใช้เป็นสื่อในการแลกเปลี่ยนคือ XML ทำให้เราสามารถเรียกใช้ component ใด ๆ ก็ได้ ใน platform ใด ๆ ก็ได้ บน protocol HTTP ซึ่งเป็น protocol สำหรับ World Wide Web อันเป็นช่องทางที่ได้รับการยอมรับทั่วโลกในการติดต่อสื่อสารกันระหว่าง application กับapplication ในปัจจุบัน

Web Service ช่วยให้การเข้าถึงข้อมูลสารสนเทศจากแอพพลิเคชันที่ต่างกันเป็นไปโดยง่าย โดยแอพพลิเคชันนั้นๆ สามารถเขียนด้วย Java และรันอยู่บน Sun Solaris Application Server หรืออาจจะเขียนด้วย C++ และรันอยู่บน Windows NT หรืออาจะเขียนด้วย Perl และรันอยู่บนเครื่อง Linux ซึ่งมาตรฐานของ Web Service ทำให้อินเทอร์เฟซของแอพพลิเคชันเหล่านี้ ถูกอธิบายโดย WSDL และทำให้อยู่ในมาตรฐานของ UDDI หลังจากนั้น จึงสามารถติดต่อสื่อสารถึงกันโดย XML ผ่าน SOAP อินเตอร์เฟซ

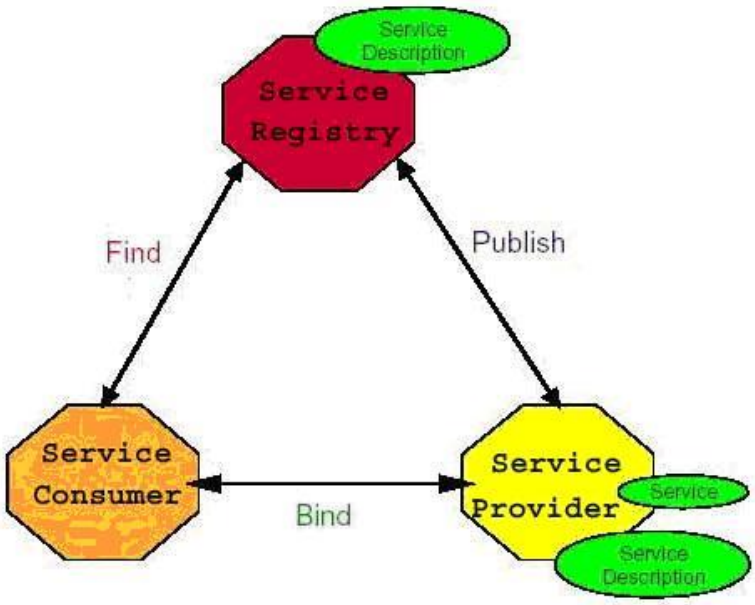
Web Service สามารถถูกเรียกใช้ภายในองค์กรเองหรือจากภายนอกองค์กร โดยผ่านไฟร์วอล์ ดังนั้นจึงมีองค์กรใหญ่ๆ มากมาย กำลังพัฒนาระบบที่มีอยู่ของตน ให้เข้ากับ Web Service ซึ่งนับเป็นการลงทุนที่คุ้มค่า เนื่องจาก Web Service สามารถเพิ่มศักยภาพในการทำงานขององค์กร อีกทั้งลดค่าใช้จ่ายในการจัดการทรัพยากรขององค์กรได้อีกทางหนึ่ง

นอกจากนั้น Web Service ยังสามารถใช้ร่วมกับ Web Application โดยส่งผ่านข้อมูลทางอินเตอร์เน็ตได้อีกด้วยซึ่งนับเป็นวิธีที่มีประสิทธิภาพในการติดต่อสื่อสารกับลูกค้าหรือหุ้นส่วน ถึงแม้จะต้องคำนึงถึงระบบรักษาความปลอดภัย และการจัดการรายการของข้อมูลอยู่ก็ตาม แต่ Web Service ได้ใช้มาตรฐานทั่วไปของ internet เรื่องดังกล่าวจึงนับเป็นเรื่องธรรมดาของการสื่อสารผ่านระบบอิเล็กทรอนิกส์

Web Services คืออะไร

- เป็นทางเลือกหนึ่งของ Connection Technology ในการ Implement SOA
- ใช้มาตรฐานและเทคโนโลยีร่วมสมัยเป็นพื้นฐาน
  - XML
  - XML Schema
  - SOAP
  - WSDL
  - UDDI
  - HTTP Transport (Default)
- มีการกำหนด Web Services Architecture และ Web Services Stack เพื่อขยายขีดความสามารถในอนาคต

“a Web Service is an application component that can access over the Web



Requestor คือ ใครก็ตามที่ต้องการเรียกใช้บริการจาก Provider ซึ่งสามารถค้นหาบริการที่ต้องการได้จาก UDDI registry หรือ Service Registry หรือติดต่อจาก Provider โดยตรง

Registry คือ ทำหน้าที่เป็นตัวกลางให้ Provider มาลงทะเบียนไว้ โดยใช้ WSDL ไฟล์ บอกรายละเอียดของบริษัทและบริการที่มีให้ ซึ่งอาจจะใช้หรือไม่ใช้ก็ได้

Provider คือ เป็นผู้ให้บริการ มีหน้าที่ในการเปิดบริการเพื่อรองรับการขอใช้บริการจาก Requestor ที่เรียกเข้ามาขอใช้

ประโยชน์ของ Web Services

1.Web Services ช่วยให้การเข้าถึงข้อมูลสารสนเทศจากแอพพลิเคชันที่ต่างกันเป็นไปโดยง่าย โดยแอพพลิเคชันนั้นๆ สามารถเขียนด้วย Java และรันอยู่บน Sun Solaris Application Server หรืออาจจะเขียนด้วย C++ และรันอยู่บน Windows NT หรืออาจะเขียนด้วย Perl และรันอยู่บนเครื่อง Linux ซึ่งมาตรฐานของ Web Service ทำให้อินเทอร์เฟซของแอพพลิเคชันเหล่านี้ ถูกอธิบายโดย WSDL และทำให้อยู่ในมาตรฐานของ UDDI หลังจากนั้น จึงสามารถติดต่อสื่อสารถึงกันโดย XML ผ่าน SOAP อินเตอร์เฟซ

2.Web Services สามารถถูกเรียกใช้ภายในองค์กรเองหรือจากภายนอกองค์กร โดยผ่านไฟร์วอล์ ดังนั้นจึงมีองค์กรใหญ่ๆ มากมาย กำลังพัฒนาระบบที่มีอยู่ของตน ให้เข้ากับ Web Services ซึ่งนับเป็นการลงทุนที่คุ้มค่า เนื่องจาก Web Services สามารถเพิ่มศักยภาพในการทำงานขององค์กร อีกทั้งลดค่าใช้จ่ายในการจัดการทรัพยากรขององค์กรได้อีกทางหนึ่ง

3.นอกจากนั้น Web Services ยังสามารถใช้ร่วมกับ Web Application โดยส่งผ่านข้อมูลทางอินเตอร์เน็ตได้อีกด้วยซึ่งนับเป็นวิธีที่มี

ประสิทธิภาพในการติดต่อสื่อสารกับลูกค้าหรือหุ้นส่วน ถึงแม้จะต้องคำนึงถึงระบบรักษาความปลอดภัย และการจัดการรายการของข้อมูลอยู่ก็ตาม แต่ Web Services ได้ใช้มาตรฐานทั่วไปของ internet เรื่องดังกล่าวจึงนับเป็นเรื่องธรรมดาของการสื่อสารผ่านระบบอิเล็กทรอนิกส์

10.Web Service สามารถทำงานกับอะไรได้บ้าง

ANS.

- Web API
- Web application
- Mobile application

การทำงานของ Web Services ประกอบไปด้วย มาตรฐานหลัก 4 อย่าง ดังนี้

1. XML (Extensible Markup Language) เป็นภาษามาตรฐานที่ทุกระบบสนับสนุน ทำให้ข้อมูลที่มีโครงสร้างของภาษา XML จะถูกนำไปประมวลผลต่ออย่างอัตโนมัติได้อย่างง่ายดาย ภาษา XML จึงถูกนำมาใช้เป็นภาษามาตรฐานในการแลกเปลี่ยนข้อมูลของ Web Services

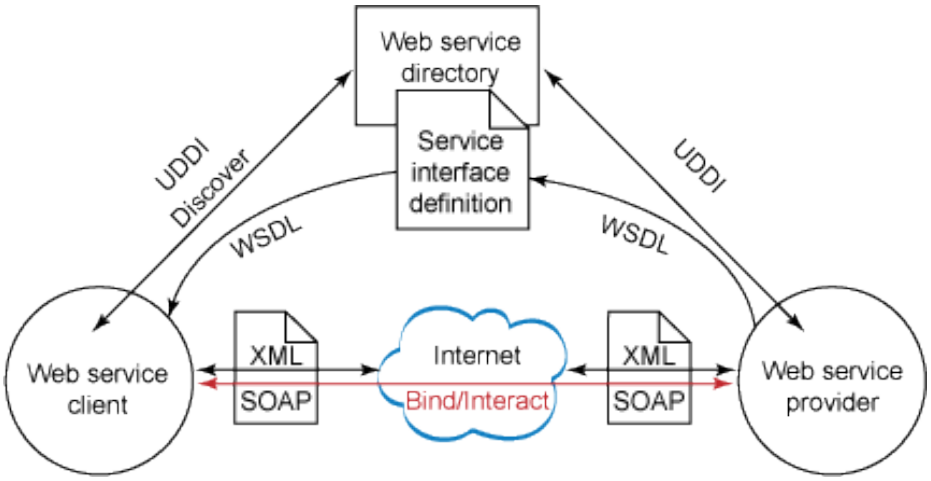
2. SOAP (Simple Object Access Protocol) เป็นมาตรฐานของเทคโนโลยี Distributed Objects โดยทำหน้าที่ส่งข้อมูลผ่านอินเทอร์เน็ตในรูปแบบของ XML ทำให้เรียกใช้งานโปรแกรมข้ามระบบผ่านทางอินเทอร์เน็ตได้

3. WSDL (Web Services Description Language) เป็นภาษามาตรฐานที่ใช้สำหรับอธิบายการใช้งานโปรแกรมที่เปิดให้บริการ ซึ่งเขียนขึ้นตามแบบมาตรฐาน XML ดังนั้น WSDL จึงเป็นเสมือนคู่มือให้กับระบบ เพื่อเรียนรู้วิธีการเรียกใช้งาน Web Services

4. UDDI (Universal Description, Discovery, and Integration) เป็นระบบมาตรฐานในการอธิบายและค้นหา Web Services โดยเป็นตัวกลางให้ provider มาลงทะเบียนไว้ โดยใช้ไฟล์ WSDL บอกรายละเอียดของบริษัทและบริการที่มีให้ ทำให้ Requestor สามารถค้นหาและทราบว่าบริษัทมีผลิตภัณฑ์และบริการอะไรบ้าง สามารถติดต่อขอดำเนินธุรกิจการค้ากับบริษัทได้โดยอัตโนมัติผ่านทาง Web Services

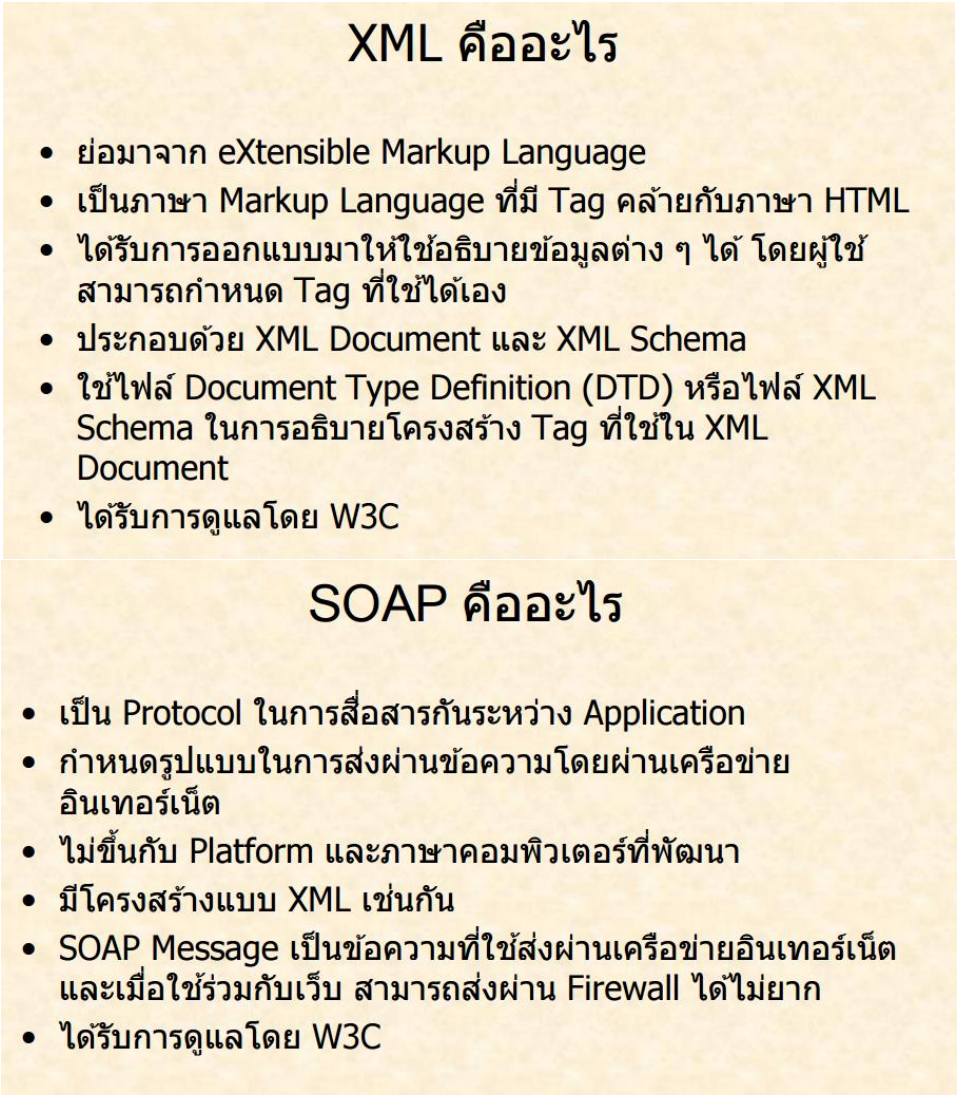
11.แผนภาพ Model Service อธิบาย

ANS.



12.ความแตกต่างของSOAP and XML

ANS.



13.REST Function ทำงานยังไง

ANS. REST หรือ Representational State Transfer มันเป็นวิธีในการสร้าง Web Service รูปแบบนึงที่อาศัย HTTP Method (GET, POST, PUT, DELETE) ในการทำงาน และส่งผลกลับมาในรูปแบบของ JSON หรือ XML ส่งผลให้สามารถรับส่งข้อมูลไปมาข้าม Platform ได้อย่างสะดวกมาก เพราะเป็นการเรียกผ่าน HTTP Protocol ที่ใช้ในการเรียกเว็บไซต์อยู่แล้ว และอีกอย่างหนึ่งที่ทำให้ REST เป็นที่นิยมคือ เรื่องของการใช้ Traffic เนื่องจากว่า เวลา REST มันส่งค่ากลับมา มันจะส่งกลับมาในรูปแบบของ JSON หรือ XML ซึ่งมีขนาดเล็ก และ Extract ออกมาใช้ได้อย่างง่ายดาย

SOAP vs. REST

A NordicAPIs infographic

NORDIC APIS

NORDICAPIS.COM

Has your boss made you responsible for your company's first API? Wondering which protocol you should use?

In this post, we'll take a fresh look at the REST vs SOAP comparison.

We've created an infographic that will show you which protocol is a better fit. We've looked at the REST vs SOAP from a use-case perspective, hopefully making it easier to choose which protocol is better suited for your job.

ORIGIN

REST

REST (Representational State Transfer) was Created in 2000 by Roy Fielding in UC, Irvine. Developed in an academic environment, this protocol embraces the philosophy of the open Web.

SOAP

SOAP (Simple Object Access Protocol), was created in 1998 by Dave Winer et al in collaboration with Microsoft. Developed by a large software company, this protocol addresses the goal of addressing the needs of the enterprise market.


BASIC
CONCEPT
REST
Makes data available as resources (nouns), for example "user" or "invoice""
SOAP
Makes data available as services (verb + noun), for example "getUser" or "PayInvoice"


PROS
XML
JSON
WSDL
REST
SOAP
• Follows the philosophy of the Open Web
• Relatively easy to implement and maintain
• Clearly separates client and server implementations
• Communication isn't controlled by a single entity
• Information can be stored by the client to prevent multiple calls
• Can return data in multiple formats (JSON, XML etc)
• Follows a formal enterprise approach
• Works on top of any communication protocol, even asynchronously
• Information about objects is communicated to clients
• Security and authorization are part of the protocol
• Can be fully described using WSDL


WHEN TO USE
REST
SOAP
• When clients and servers operate on a Web environment
• When information about objects doesn't need to be communicated to the client
• When clients need to have access to objects available on servers
• When you want to enforce a formal contract between client and server


WHEN NOT TO USE
REST
SOAP
• When you need to enforce a strict contract between client and server
• When performing transactions that involve multiple calls
• When you want the majority of developers to easily use your API
• When your bandwidth is very limited

COMMON USE CASES

REST

- Social Media services
- Social Networks
- Web Chat services
- Mobile Services

SOAP

- Financial services
- Payment gateways
- Telecommunication services

POPULAR EXAMPLES

REST

- Twitter API
- LinkedIn API
- Slack API

SOAP

- Salesforce SOAP API
- Paypal SOAP API
- Clickatell SMS SOAP API