

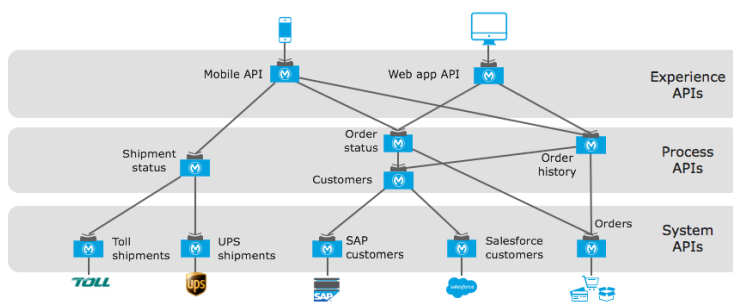


Module 1: Introducing Application Networks and API-Led Connectivity

At the end of this module, you should be able



- Explain what an application network is and its benefits
- Describe how to build an application network using API-led connectivity
- Explain what web services and APIs are
- Explore API directories and references
- Make calls to secure and unsecured APIs

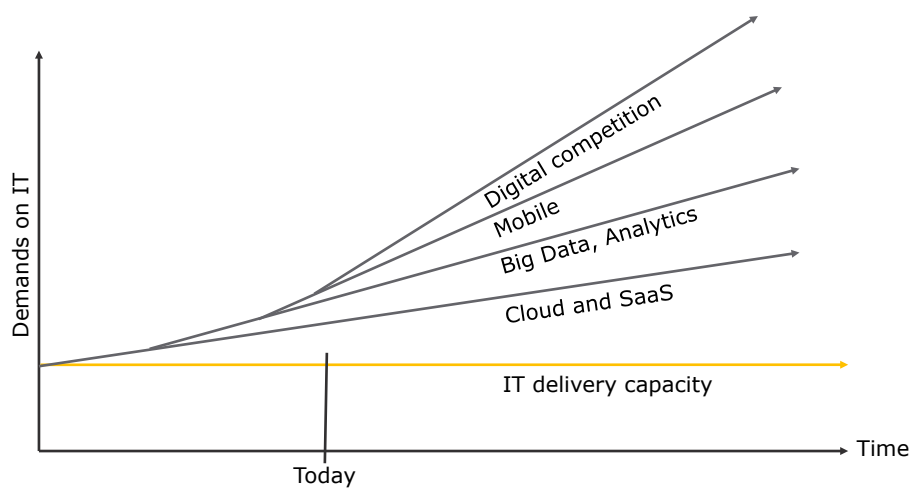


All contents © MuleSoft Inc.

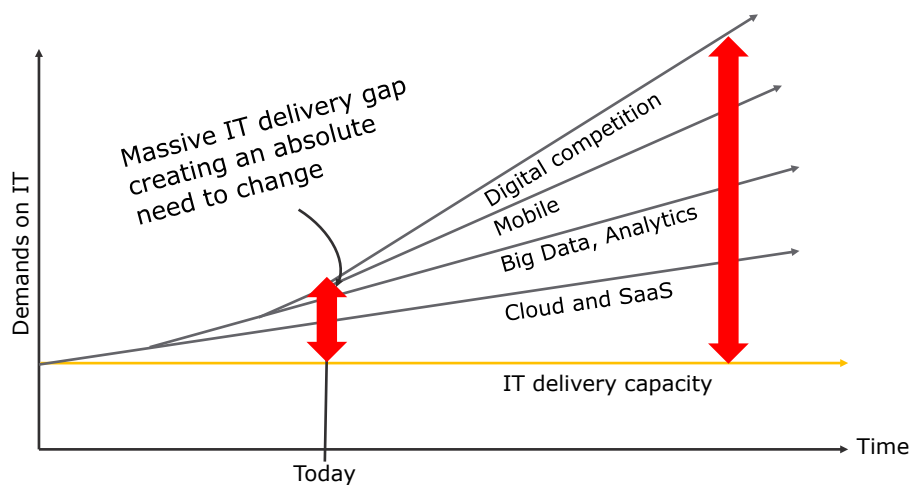
Identifying the problems faced by IT today



Biggest challenge: IT cannot go fast enough



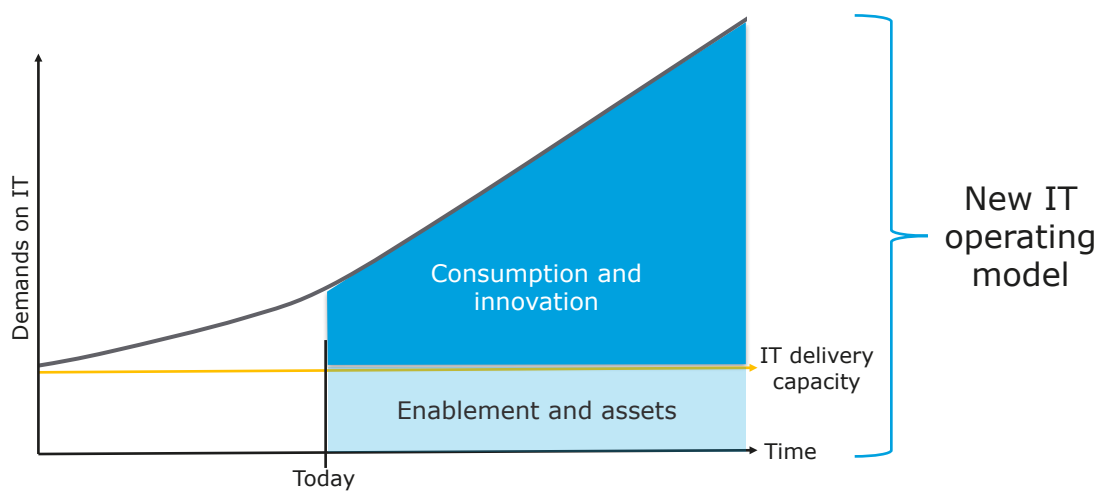
Digital pressures create a widening IT delivery gap



All contents © MuleSoft Inc.

5

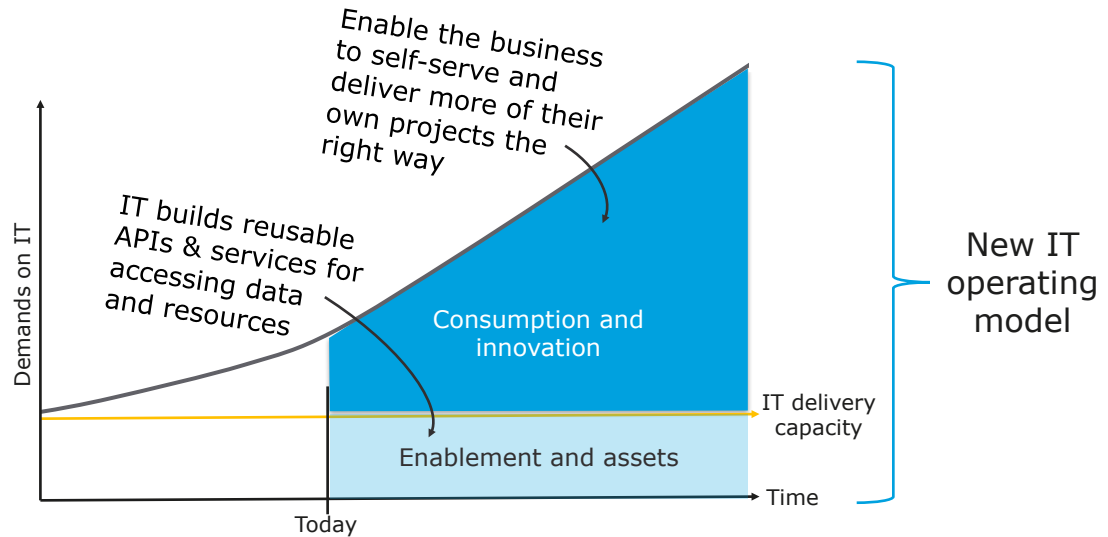
A new way of working to close the delivery gap



All contents © MuleSoft Inc.

6

A new way of working to close the delivery gap



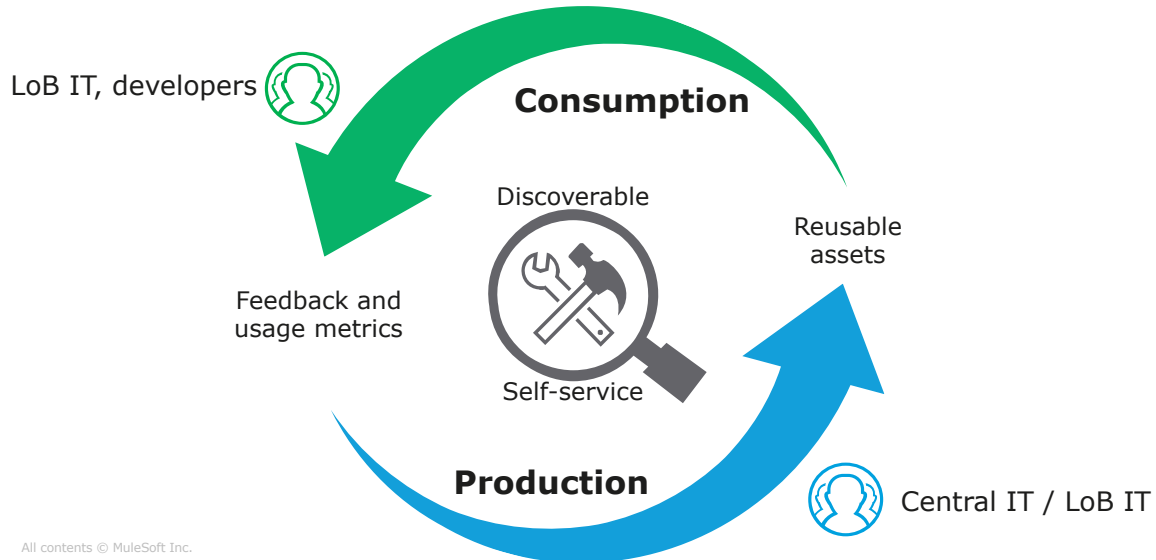
All contents © MuleSoft Inc.

7

Introducing a new IT operating model



New operating model emphasizes consumption



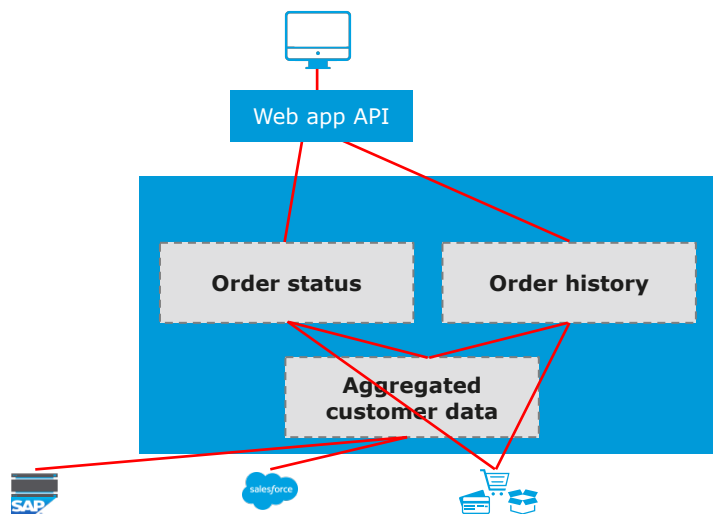
9

A common project-based approach



Project objective: Web app provides real-time order status and order history for sales team engaging with customers

- Order data in eCommerce system
- Inventory data in SAP
- Customer data in SAP, Salesforce



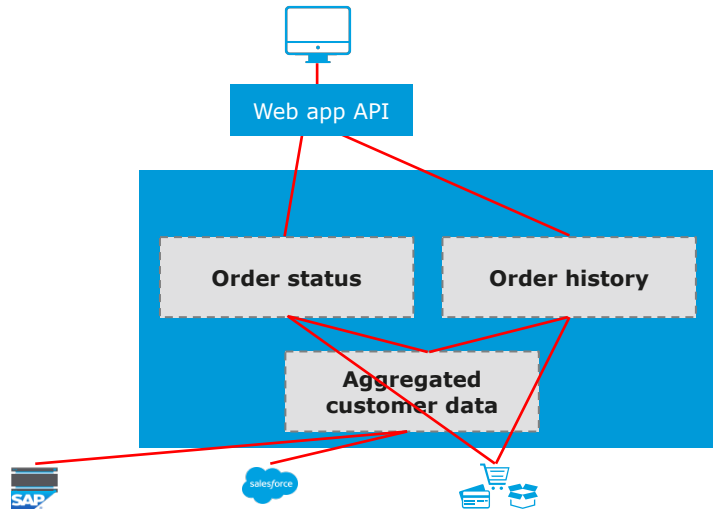
All contents © MuleSoft Inc.

10

A common project-based approach



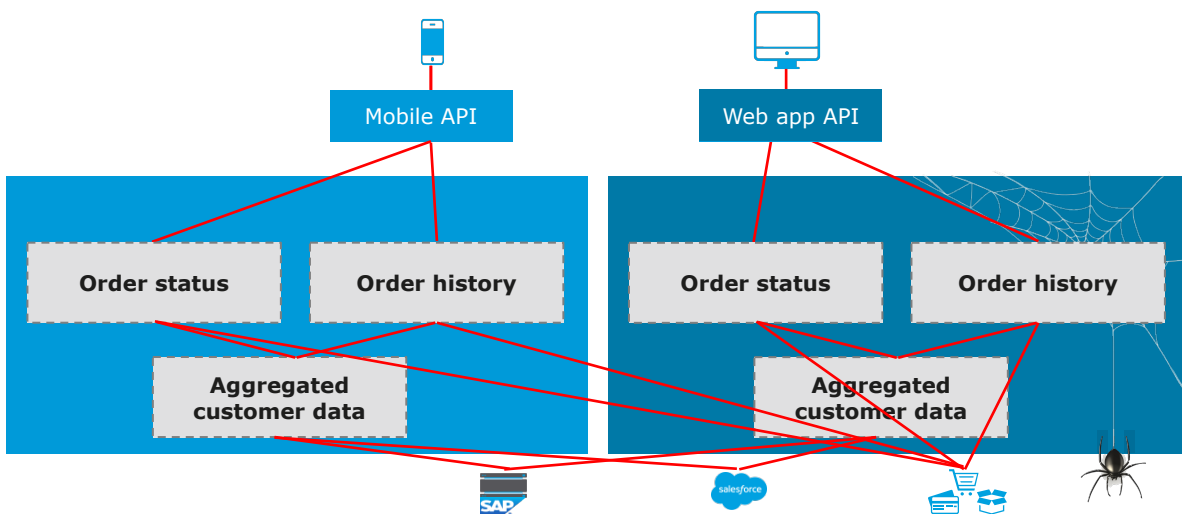
- ✓ On time and within budget
- ✗ Limited opportunity for reuse
- ✗ Tight coupling = brittleness
- ✗ Difficult to govern
- ⚠ Meets business requirements



All contents © MuleSoft Inc.

11

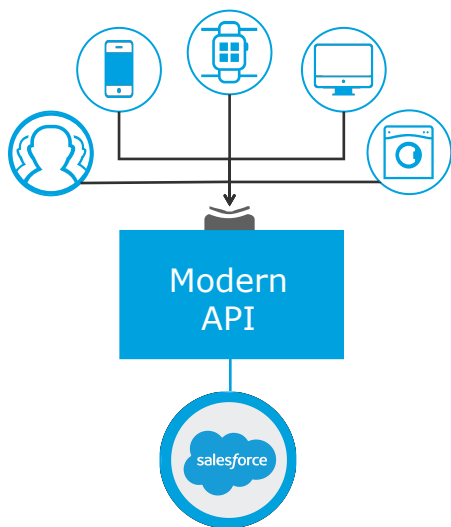
6 months later...



All contents © MuleSoft Inc.

12

Modern API: The core enabler of a new operating model

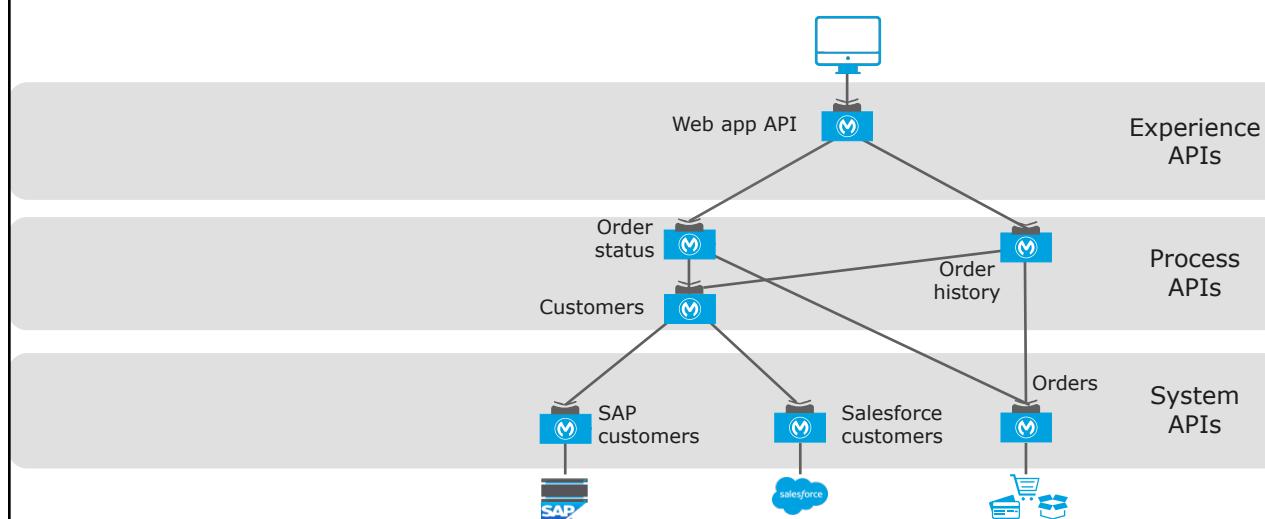


- Discoverable and accessible through self-service
- Productized and designed for ease of consumption
- Easily managed for security, scalability, and performance

All contents © MuleSoft Inc.

13

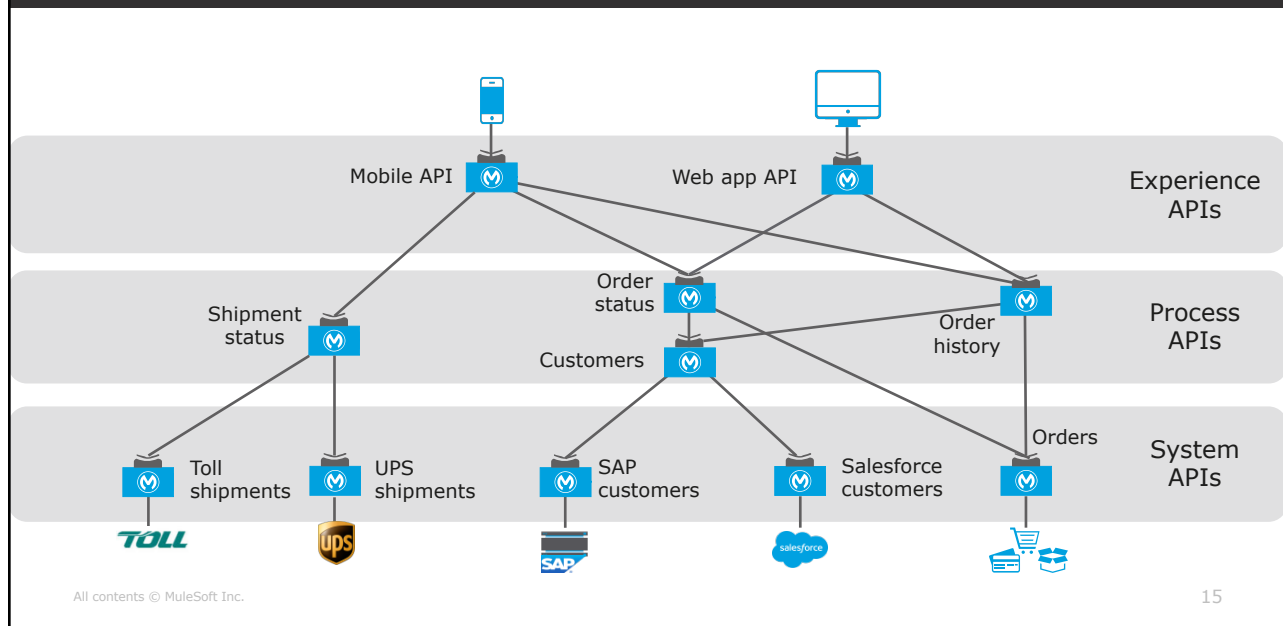
The API-led connectivity approach



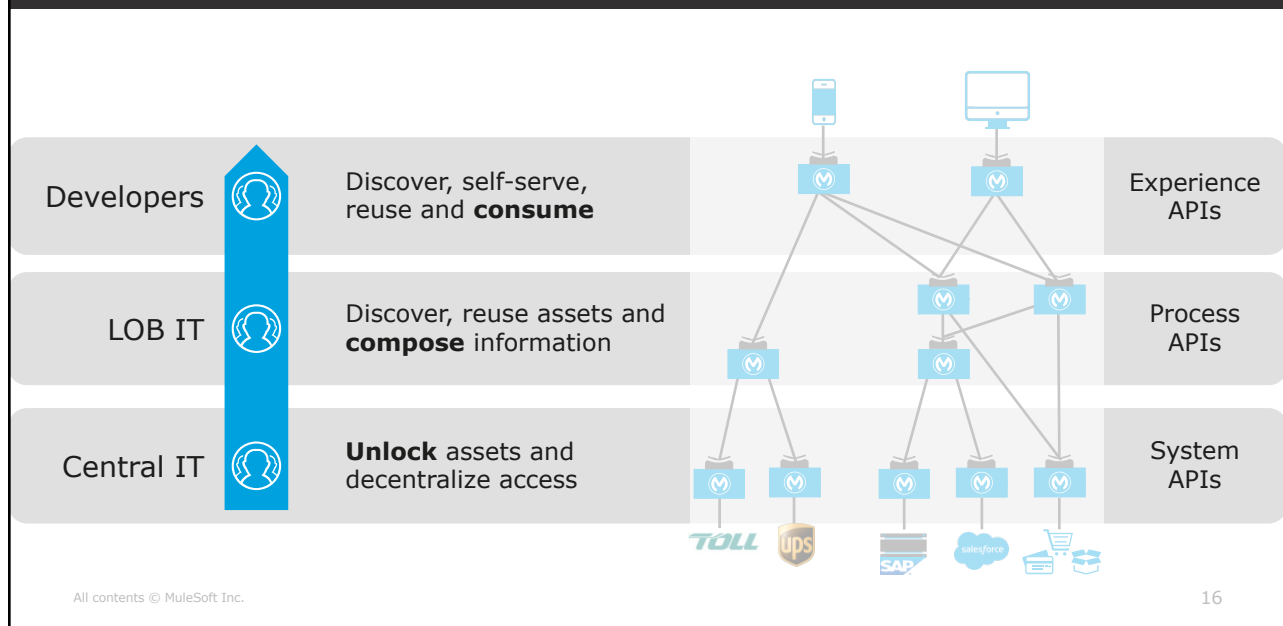
All contents © MuleSoft Inc.

14

The API-led connectivity approach



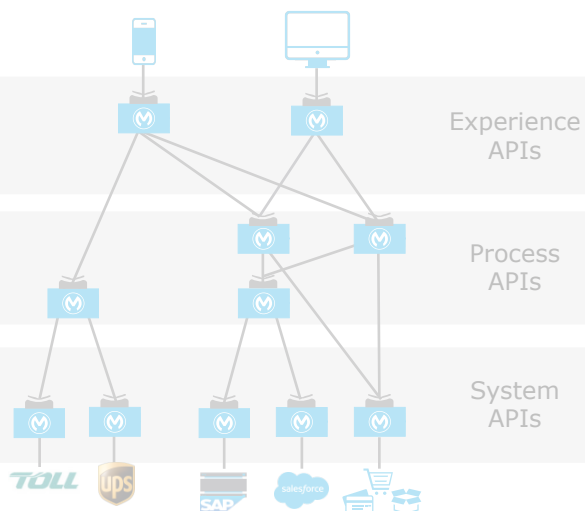
Enable and empower the entire organization



Drive outcomes with API-led connectivity



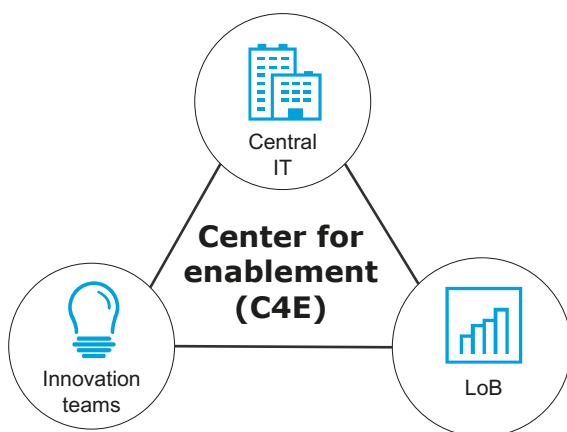
- ✓ On time and within budget
- ✓ Drives reuse vs build
- ✓ Designs in readiness for change
- ✓ Builds in governance, compliance, security, and scalability
- ✓ Meets the needs of your business



All contents © MuleSoft Inc.

17

C4E: Organizing differently to drive API-led connectivity



- C4E is a cross functional team
- C4E ensures that assets are
 - Productized and published
 - Consumable
 - Consumed broadly
 - Fully leveraged
- Success of C4E measured on asset consumption

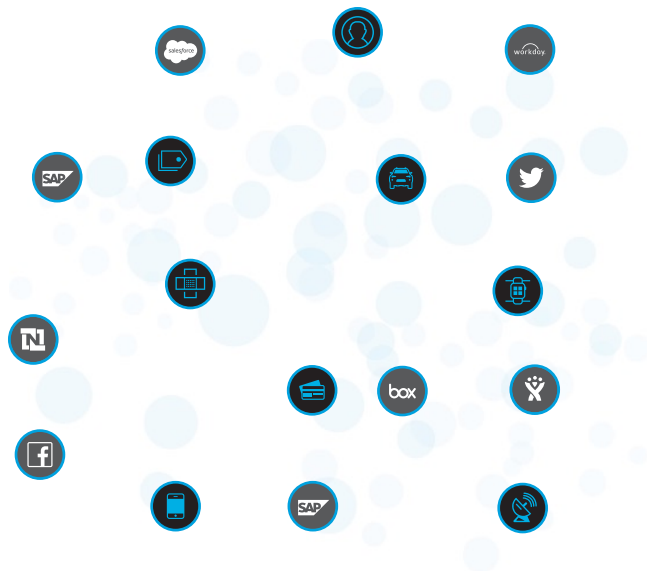
All contents © MuleSoft Inc.

18

Achieving an application network



Application landscape



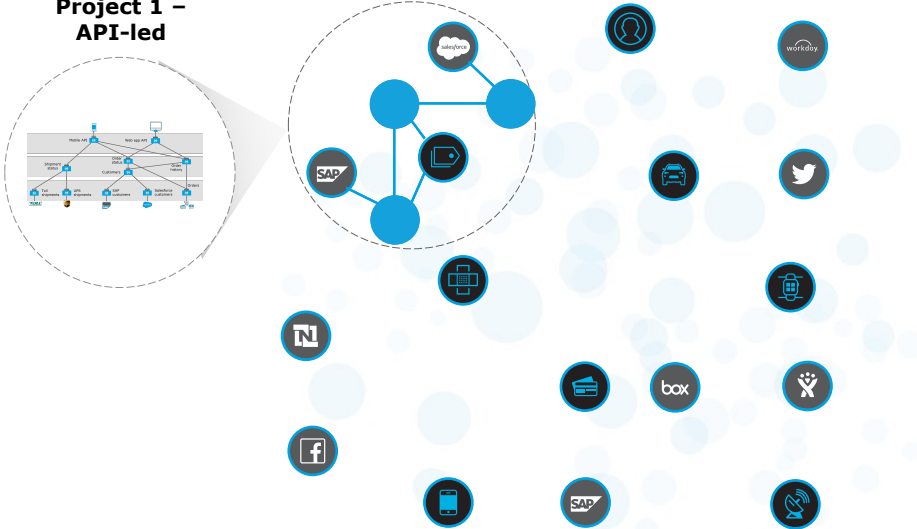
All contents © MuleSoft Inc.

20

Every project adds value to the application network



Project 1 – API-led



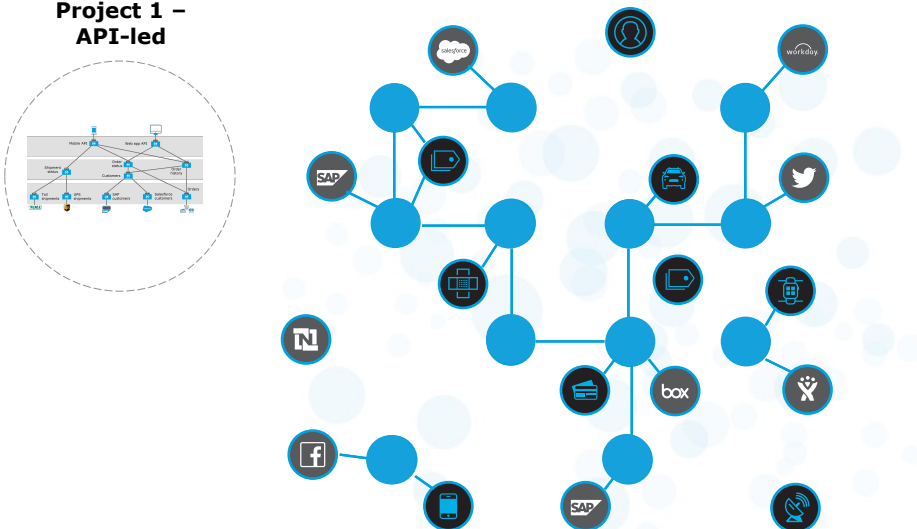
All contents © MuleSoft Inc.

21

Every project adds value to the application network



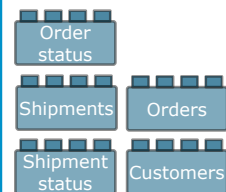
Project 1 – API-led



All contents © MuleSoft Inc.

C4E

Self-serve assets
on the
application network

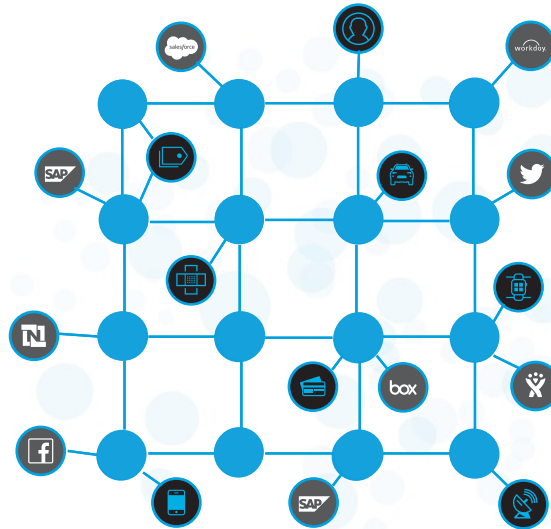
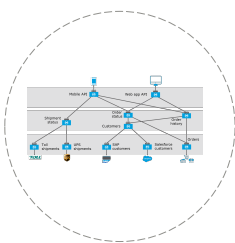


22

Every project adds value to the application network

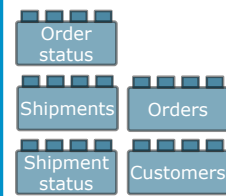


Project 1 – API-led



C4E

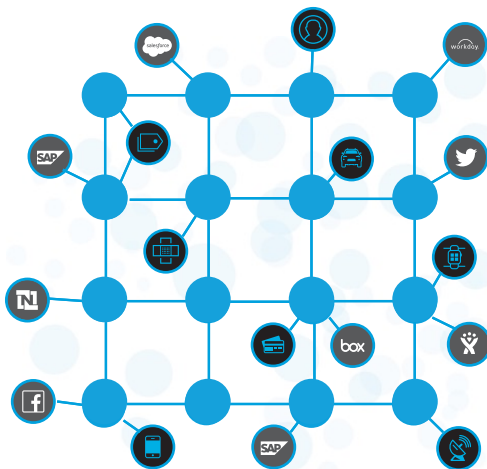
Self-serve assets on the application network



All contents © MuleSoft Inc.

23

Speed. Agility. Innovation.



An application network

- Emerges bottoms-up via self-service
- Provides visibility, security and governability at every API node
- Is recomposable: it bends, not breaks – **built for change**

All contents © MuleSoft Inc.

24

Deconstructing APIs



What exactly is an API?



- An **API** is an **A**pplication **P**rogramming **I**nterface
- It provides the information for how to communicate with a software component, defining the
 - Operations (what to call)
 - Inputs (what to send with a call)
 - Outputs (what you get back from a call)
 - Underlying data types
- It defines functionalities independent of implementations
 - You can change what's going on behind the scenes without changing how people call it

What do people mean when they say API?



They could be referring to a number of things...

1. An API interface definition file

- Defines what you can call, what you send it, and what you get back

2. A web service

- The actual API implementation you can make calls to or the interface of that API implementation

3. An API proxy

- An application that controls access to a web service, restricting access and usage through the use of an API gateway

All contents © MuleSoft Inc.

27

Reviewing web services



What is a web service?



- Different software systems often need to exchange data with each other
 - Bridging protocols, platforms, programming languages, and hardware architectures
- A **web service** is a method of communication that allows two software systems to exchange data over the internet
- Systems interact with the web service in a manner prescribed by some defined rules of communication
 - How one system can request data from another system, what parameters are required, the structure of the return data, and more

All contents © MuleSoft Inc.

29

The parts of a web service



- **The web service API**
 - Describes how you interact with the web service
 - It may or may not (though it should!) be explicitly defined in a file
 - It could be any sort of text in any type of file but ideally should implement some standard API description language (or specification)
- **The web service interface implementing the API**
 - Is the code providing the structure to the application so it implements the API
 - This may be combined with the actual implementation code
- **The web service implementation itself**
 - Is the actual code and application

All contents © MuleSoft Inc.

30

Two main types of web services



- SOAP web services
 - Traditional, more complex type
 - The communication rules are defined in an XML-based WSDL (Web Services Description Language) file
- **RESTful web services**
 - Recent, simpler type based on representational state transfer (REST) based communications
 - Use the existing HTTP communication protocol

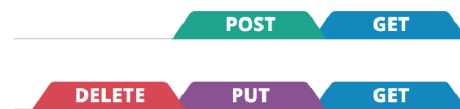
Reviewing RESTful web services



RESTful web services



- REST stands for **R**epresentational **S**tate **T**ransfer
 - An architectural style where clients and servers exchange representations of resources using standard HTTP protocol
- Other systems interact with the web service using the HTTP protocol
 - The HTTP request method indicates which operation should be performed on the object identified by the URL
 - GET, POST, DELETE, PUT, PATCH



All contents © MuleSoft Inc.

33

RESTful web service requests



- Data and resources are represented using URIs
- Resources are accessed or changed using a fixed set of operations
 - **GET** retrieves the current state of a resource in some representation (usually JSON or XML)
 - **POST** creates a new resource
 - **DELETE** deletes a resource
 - **PUT** replaces a resource completely
 - If the resource doesn't exist, a new one is created
 - **PATCH** partially updates a resource
 - Just submitted data



All contents © MuleSoft Inc.

34

Example RESTful web service calls



- (GET)/companies
- (GET)/companies?country=France
- (GET)/companies/3
- (POST)/companies with JSON/XML in HTTP body
- (DELETE)/companies/3
- (PUT)/companies/3 with JSON/XML in HTTP body

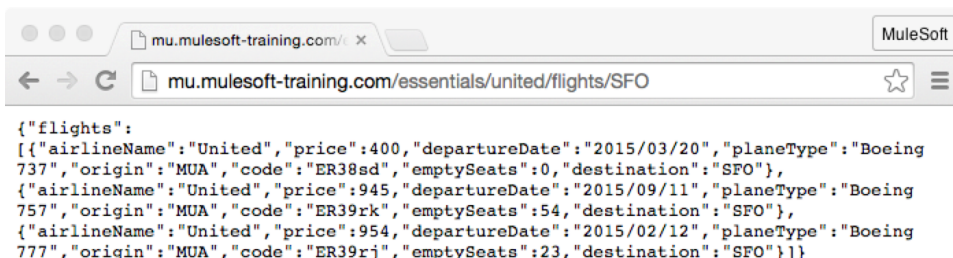
All contents © MuleSoft Inc.

35

Example RESTful web service response



- JSON (JavaScript Object Notation)
 - A lightweight data-interchange format (without a lot of extra XML markup)
 - Human-readable results (usually JSON or XML)
 - Supports collections and maps



```
{
  "flights": [
    {
      "airlineName": "United",
      "price": 400,
      "departureDate": "2015/03/20",
      "planeType": "Boeing 737",
      "origin": "MUA",
      "code": "ER38sd",
      "emptySeats": 0,
      "destination": "SFO"
    },
    {
      "airlineName": "United",
      "price": 945,
      "departureDate": "2015/09/11",
      "planeType": "Boeing 757",
      "origin": "MUA",
      "code": "ER39rk",
      "emptySeats": 54,
      "destination": "SFO"
    },
    {
      "airlineName": "United",
      "price": 954,
      "departureDate": "2015/02/12",
      "planeType": "Boeing 777",
      "origin": "MUA",
      "code": "ER39rj",
      "emptySeats": 23,
      "destination": "SFO"
    }
  ]
}
```

All coi

36

Walkthrough 1-1: Explore an API directory and an API reference



- Browse the ProgrammableWeb API directory
- Explore the API reference for the Twitter API

Developer Use cases Products Docs More Apply Search Sign In

Search all documentation...

Post, retrieve and engage with Tweets

Overview Guides API Reference

The following API endpoints can be used to programmatically create, retrieve and delete Tweets, Retweets and Likes:

Tweets	Retweets	Likes (formerly favorites)
<ul style="list-style-type: none"> • POST statuses/update • POST statuses/destroy/id • GET statuses/show/id 	<ul style="list-style-type: none"> • POST statuses/retweet/id • POST statuses/unretweet/id • GET statuses/retweets/id • GET statuses/retweets_of_me • GET statuses/retweeters/ds 	<ul style="list-style-type: none"> • POST favorites/create/id • POST favorites/destroy/id • GET favorites/list

37

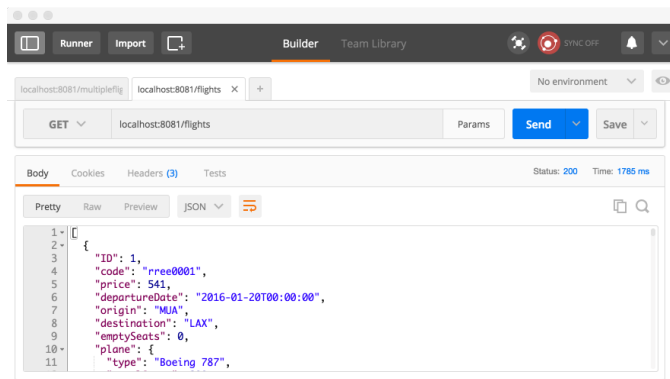
Calling RESTful web services



Calling RESTful web services



- To call web services, you need to write code or have a tool to make the HTTP requests
 - Need to be able to specify the HTTP method, request headers, and request body
 - Postman, A cURL command-line utility, Advanced Rest Client (for Chrome) +



All contents © MuleSoft Inc.

33

Making calls to RESTful APIs



- **Unsecured APIs**
 - The API may be public and require no authentication
- **Secured APIs**
 - The API may be secured and require authentication
 - You may need to provide credentials and/or a token
 - Often a proxy is created to govern access to an API
 - We will call and then later create an API secured by credentials
 - You can also secure an API with other authentication protocols
 - OAuth, SAML, JWT, and more

All contents © MuleSoft Inc.

40

Getting responses from web service calls



- RESTful web services return an HTTP status code with the response
- The status code provides client feedback for the outcome of the operation (succeeded, failed, updated)
 - A good API should return status codes that align with the HTTP spec

Response

200

401

Type application/atom+xml

Schema

Examples

☒

```
<?xml version='1.0' encoding='UTF-8'?>
<feed xmlns='http://www.w3.org/2005/Atom'
      xmlns:openSearch='http://a9.com/~spec/opensearch/1.1/'
      xmlns:gContact='http://schemas.google.com/contact/2008'
      xmlns:batch='http://schemas.google.com/gdata/batch'
      xmlns:gd='http://schemas.google.com/g/2005'
      gd:etag='feedEtag'>
```

All contents © MuleSoft Inc.

41

Common HTTP status codes



Code	Definition	Returned by
200	OK – The request succeeded.	GET, DELETE, PATCH, PUT
201	Created – A new resource or object in a collection.	POST
304	Not modified – Nothing was modified by the request.	PATCH, PUT
400	Bad request – The request could not be performed by the server due to bad syntax or other reason in request.	All
401	Unauthorized – Authorization credentials are required or user does not have access to the resource/method they are requesting.	All
404	Resource not found – The URI is not recognized by the server.	All
500	Server error – Generic something went wrong on the server side.	All

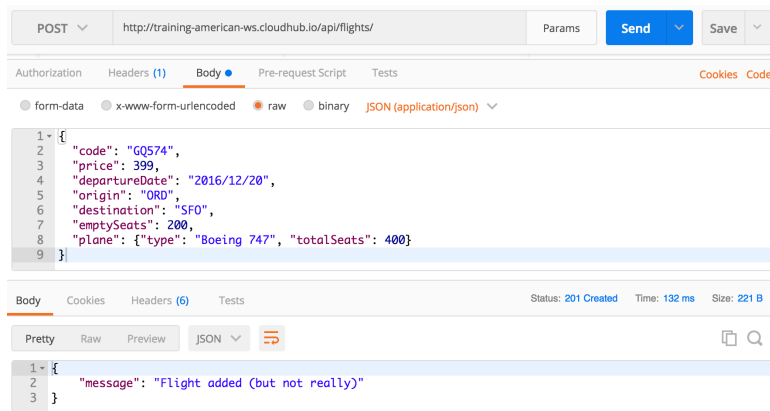
All contents © MuleSoft Inc.

42

Walkthrough 1-2: Make calls to an API



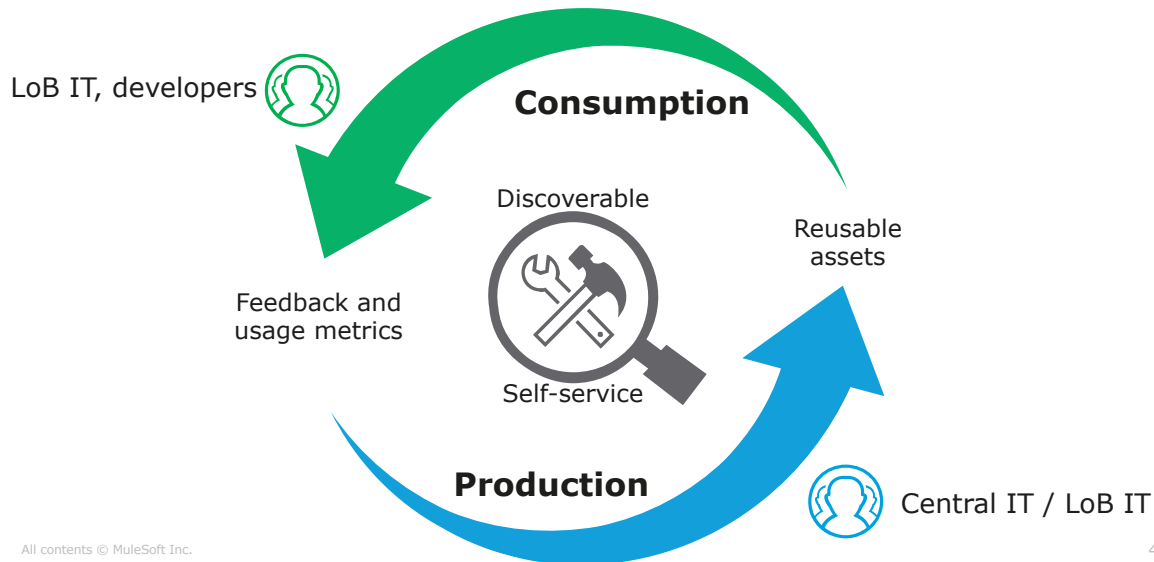
- Use Postman to make calls to an unsecured API (an implementation)
- Make GET, DELETE, POST, and PUT calls
- Use Postman to make calls to a secured API (an API proxy)



Successfully creating
application networks using
API-led connectivity



Producing discoverable and consumable assets is key



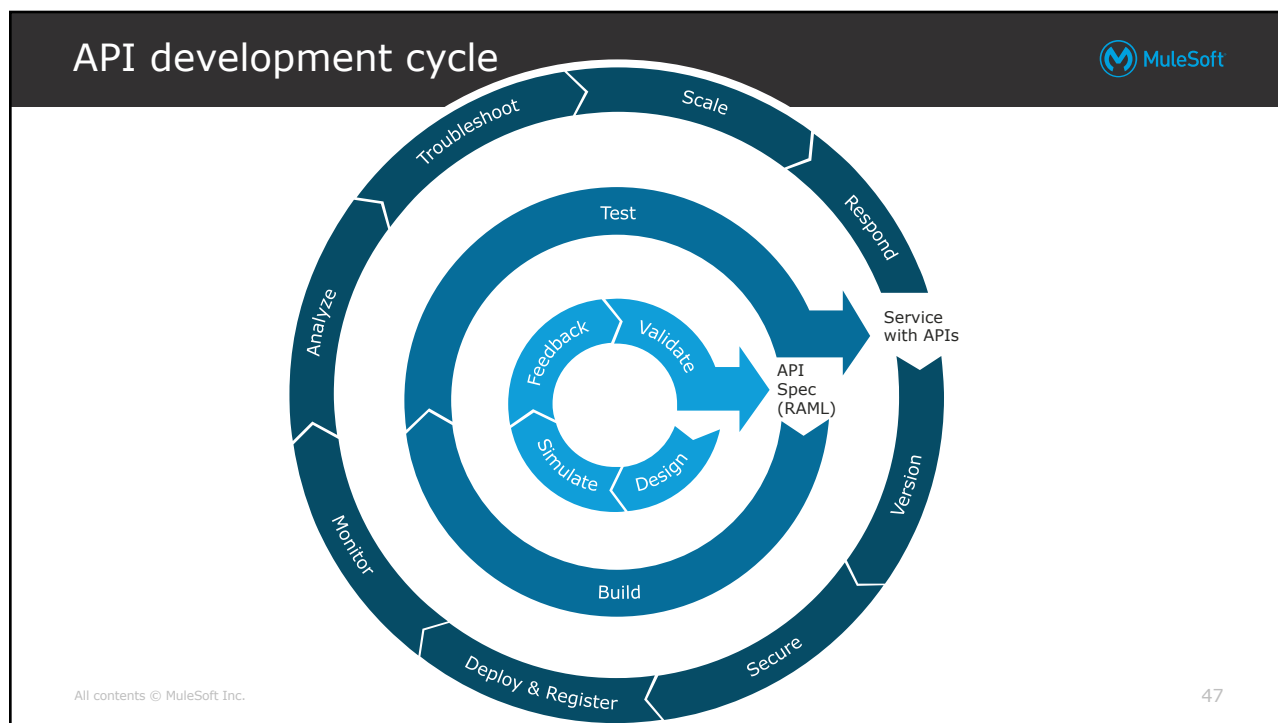
Designing for API success



- Create APIs that developers **can find** and **want to use** and share with others
 - Design the API for the business use cases it will fulfill, not to model the backend services or applications they expose
 - Focus on performance of client applications and user experience
- Take an **API design-first approach!**
- **Get API design right** before investing in building it
 - Define it iteratively getting feedback from developers on its usability and functionality along the way
 - Building the implementation of an API is time consuming and expensive to undo

All contents © MuleSoft Inc.

46



Summary



Summary



- Companies today need to **rapidly adopt and develop** new technologies in order to stay relevant to customers & keep competitive
- IT needs to be able to rapidly integrate resources and make them **available for consumption**
 - An **API-led connectivity** approach can help achieve this
- To drive API-led connectivity, create a **C4E** (Center for Enablement)
 - A cross-functional team to ensure assets across the organization are productized, published, and widely consumed
- **An application network** is a network of applications, data, and devices connected with APIs to make them pluggable and to create reusable services

All contents © MuleSoft Inc.

49

Summary



- A **web service** is a method of communication that allows two software systems to exchange data over the internet
- An **API** is an application programming interface that provides info for how to communicate with a software component
- The **term API** is often used to refer to any part of RESTful web service
 - The web service API (definition file)
 - The web service interface implementing the API
 - The web service implementation itself
 - A proxy for the web service to control access to it
- **RESTful** web services use standard HTTP protocol and are easy to use
 - The HTTP request method indicates which operation should be performed on the object identified by the URL

All contents

50