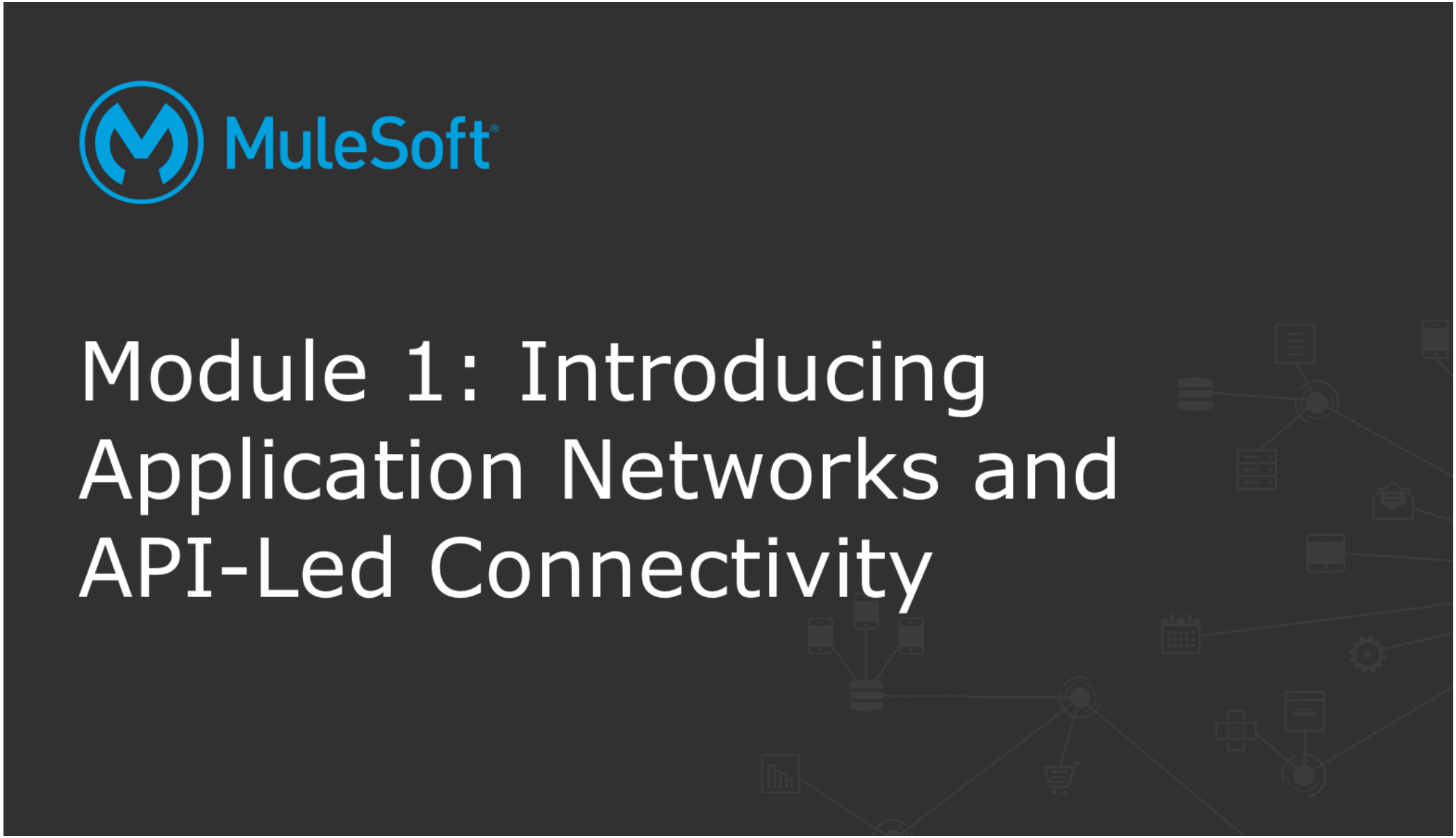
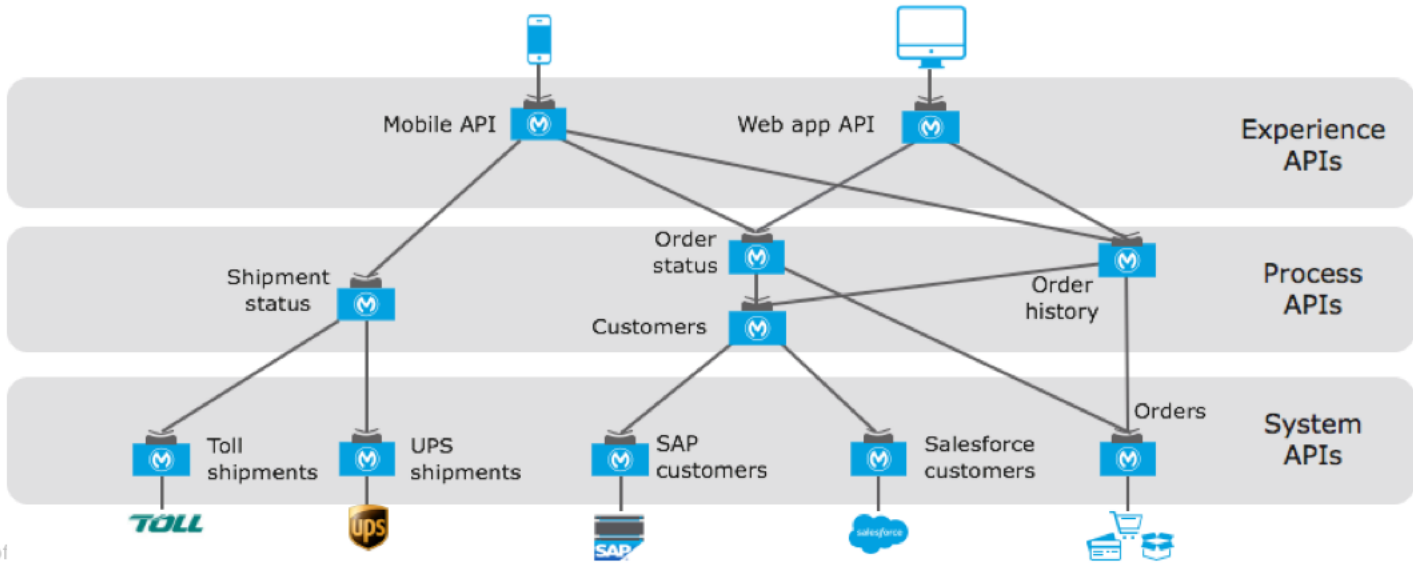
hello


MuleSoft
Module 1: Introducing Application Networks and API-Led Connectivity

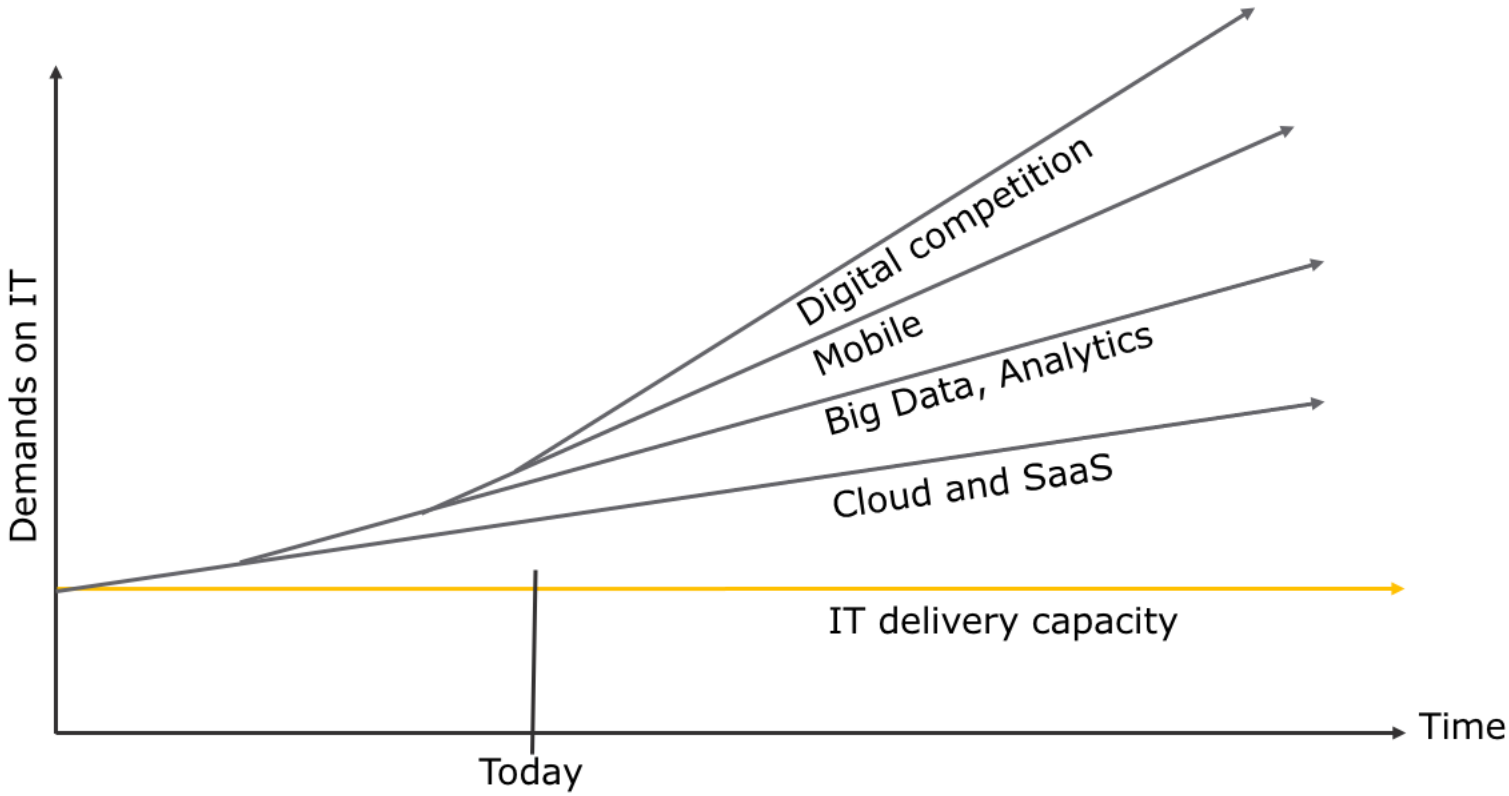
At the end of this module, you should be able

- 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
- Make calls to secure and unsecured APIs

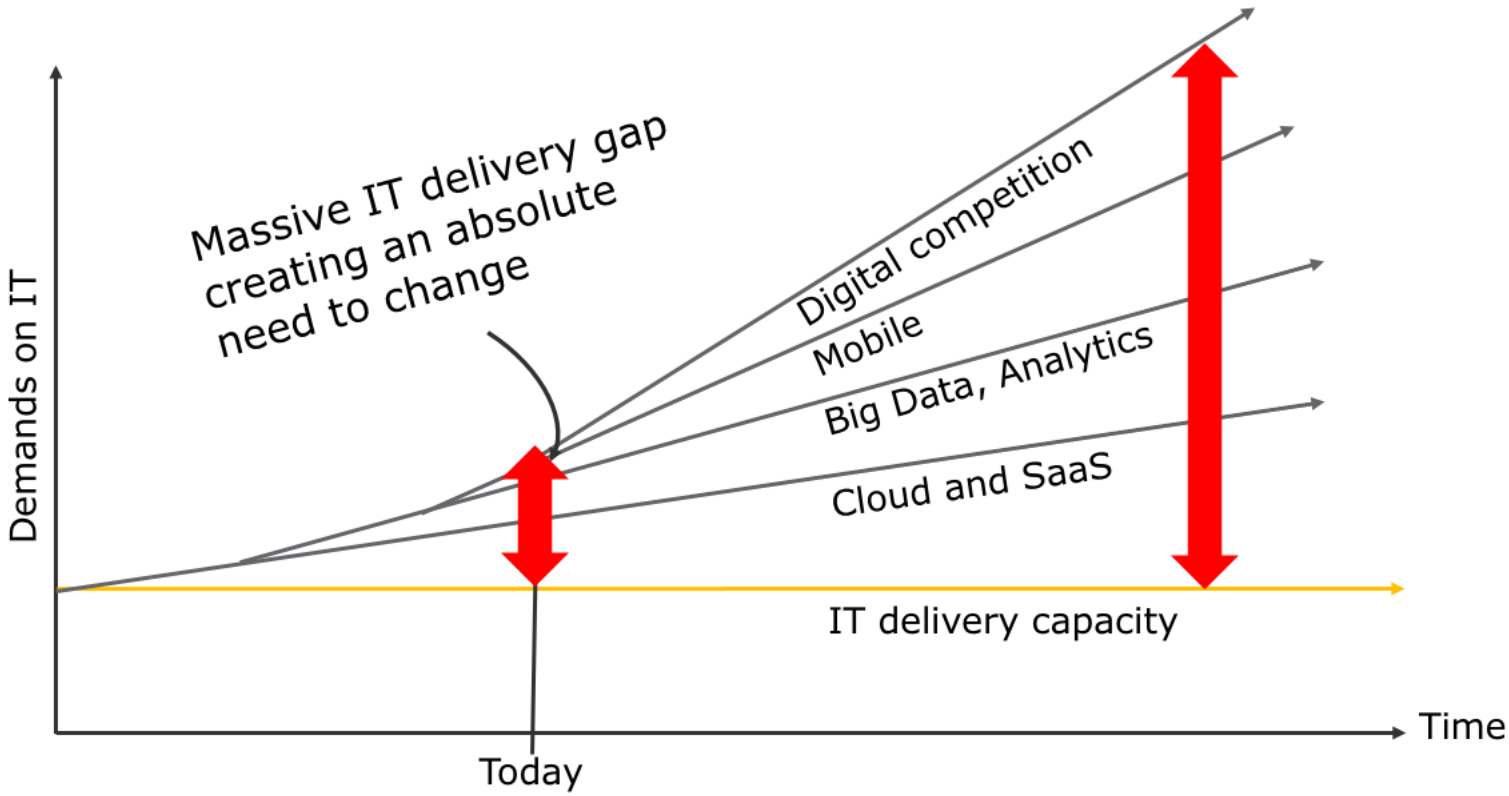


Identifying the problems faced by IT today

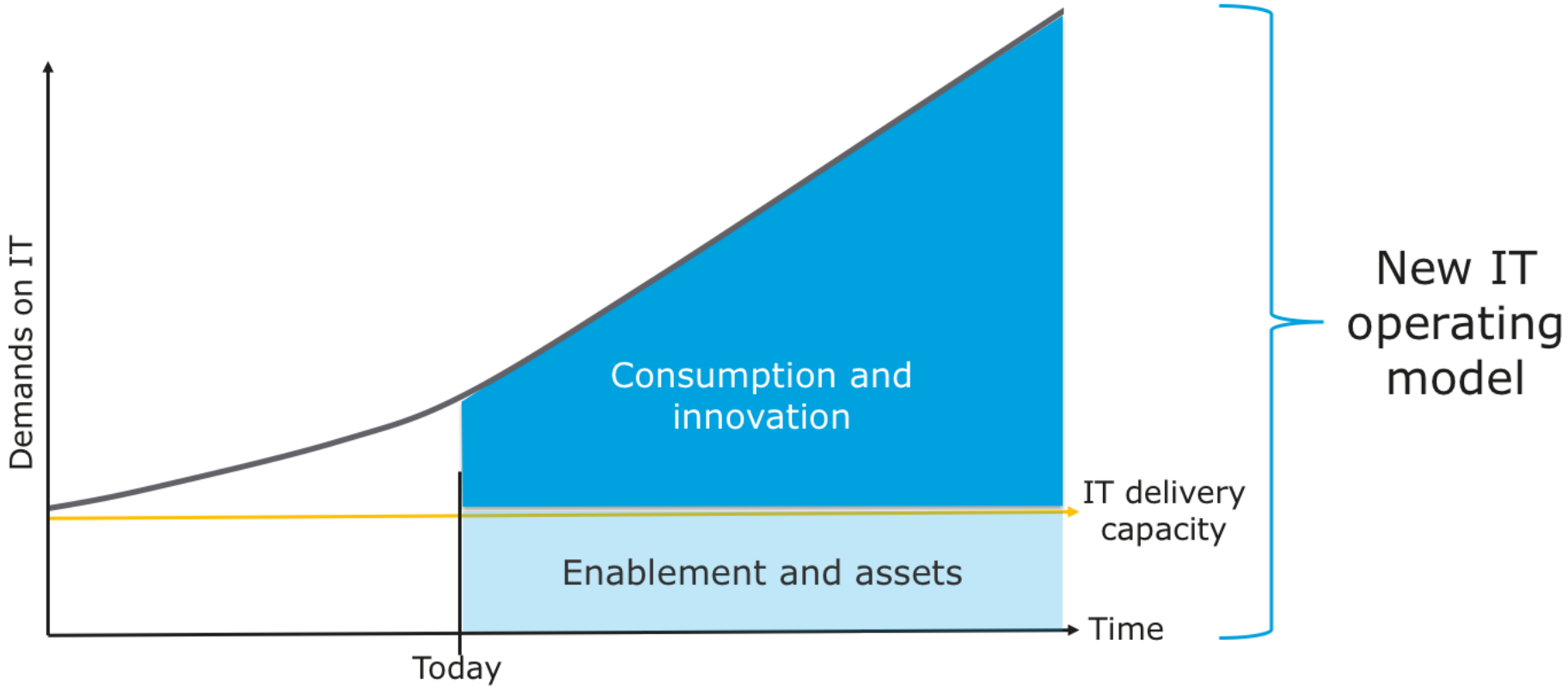
Biggest challenge: IT cannot go fast enough



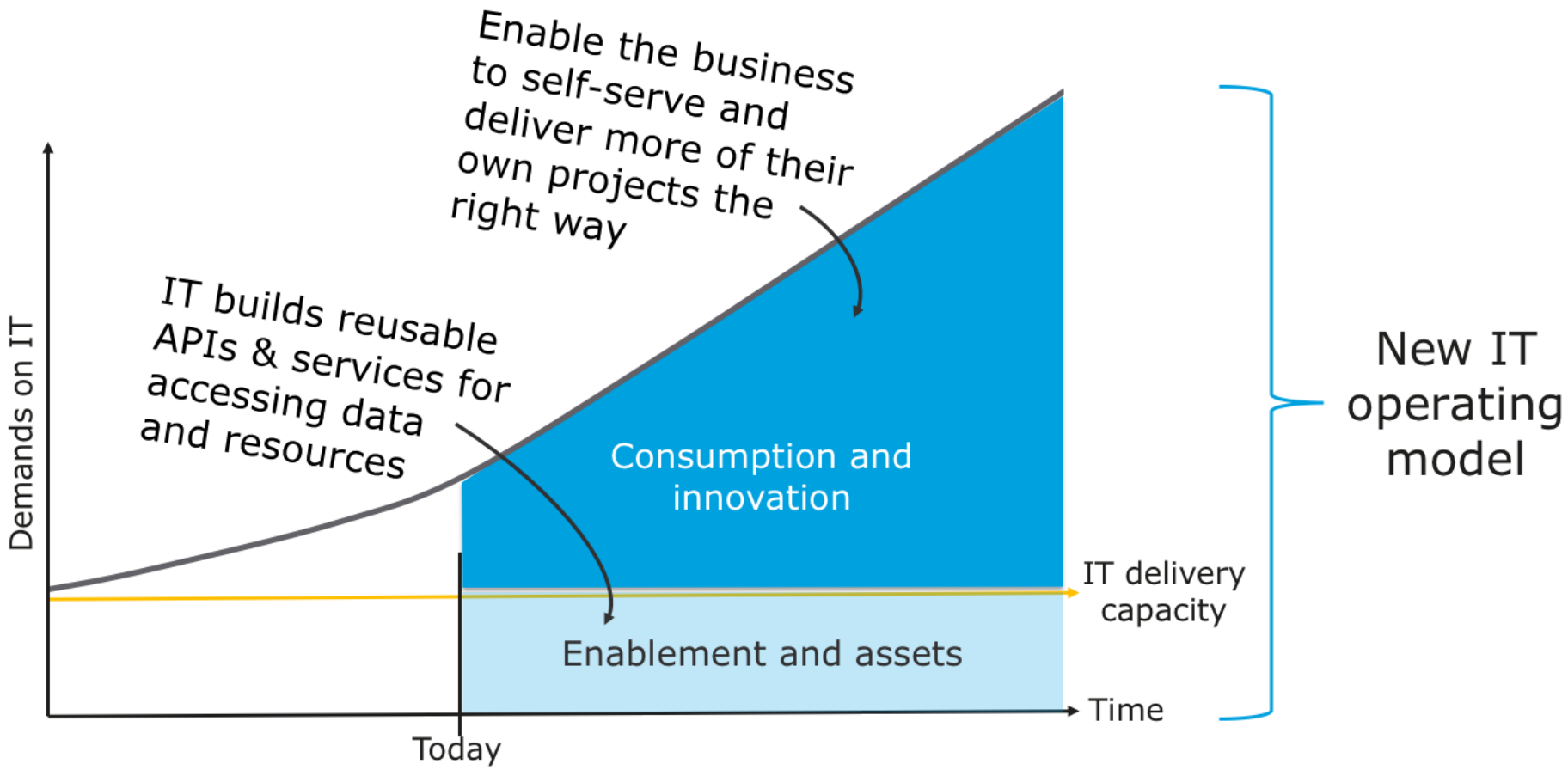
Digital pressures create a widening IT delivery gap



A new way of working to close the delivery gap

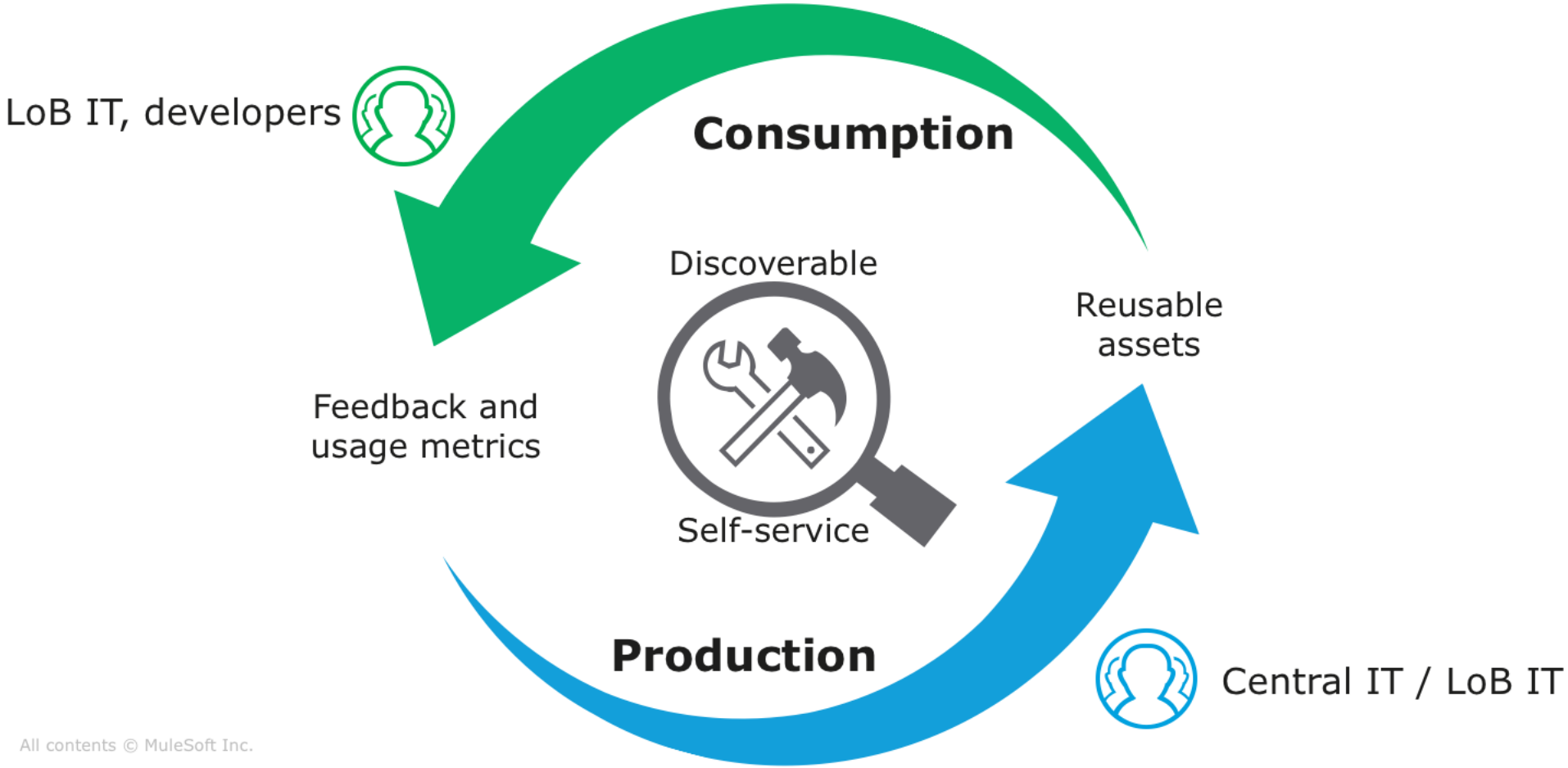


A new way of working to close the delivery gap



Introducing a new IT operating model

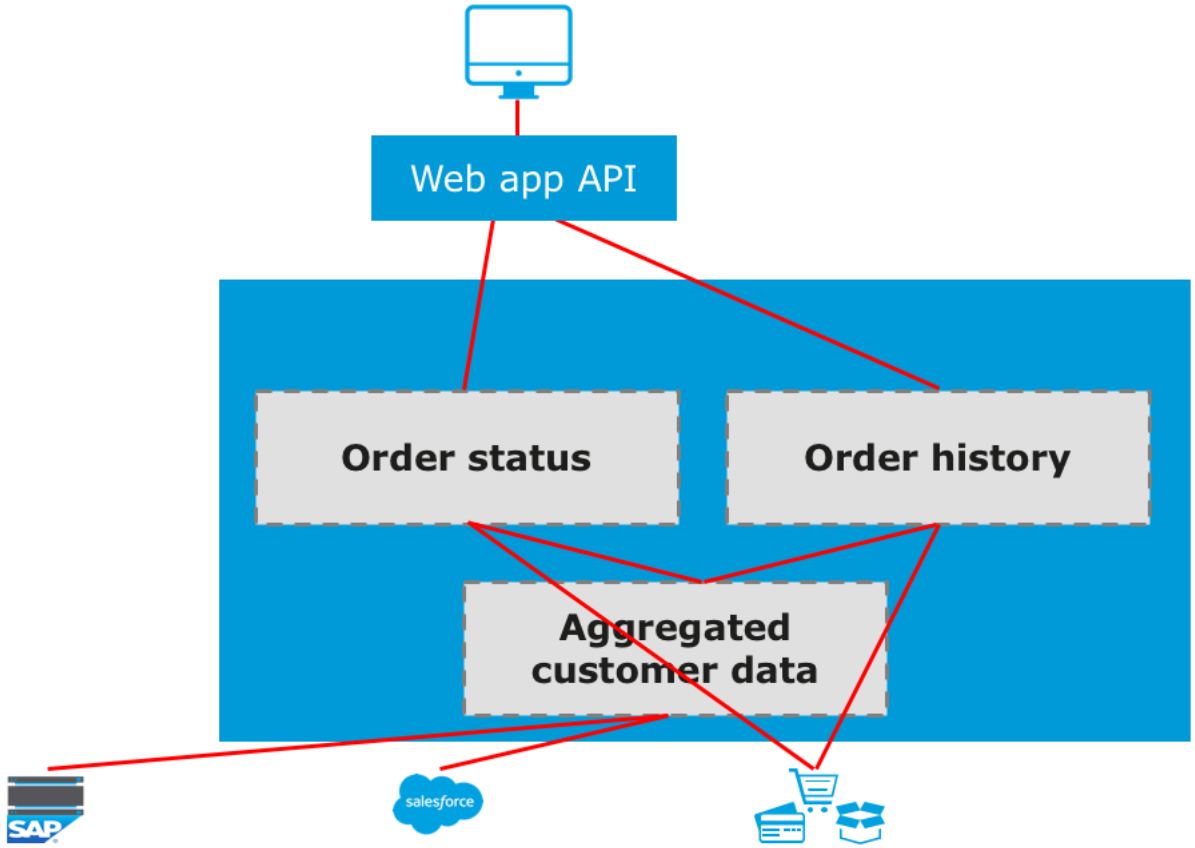
New operating model emphasizes consumption



A common project-based approach

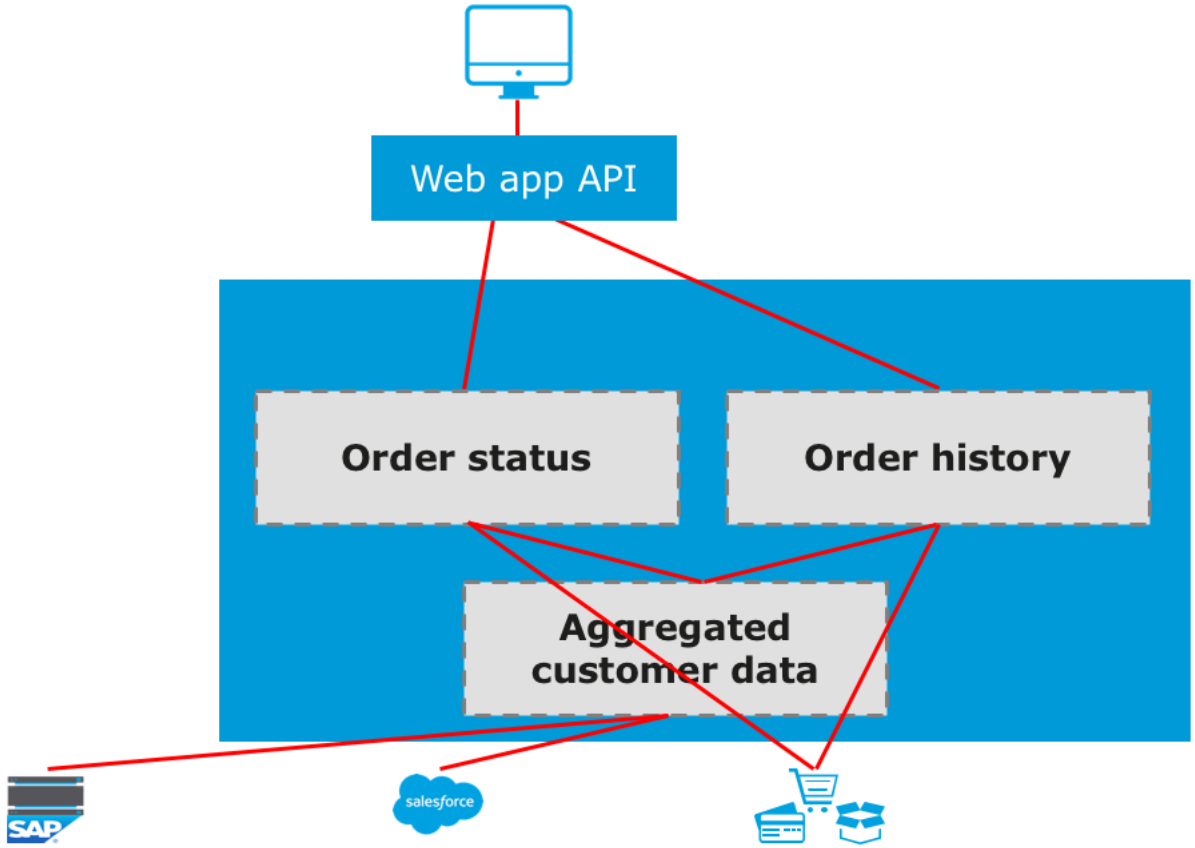
Project objective: Web app provides real-time order status and order history for sales team engaging with customers

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

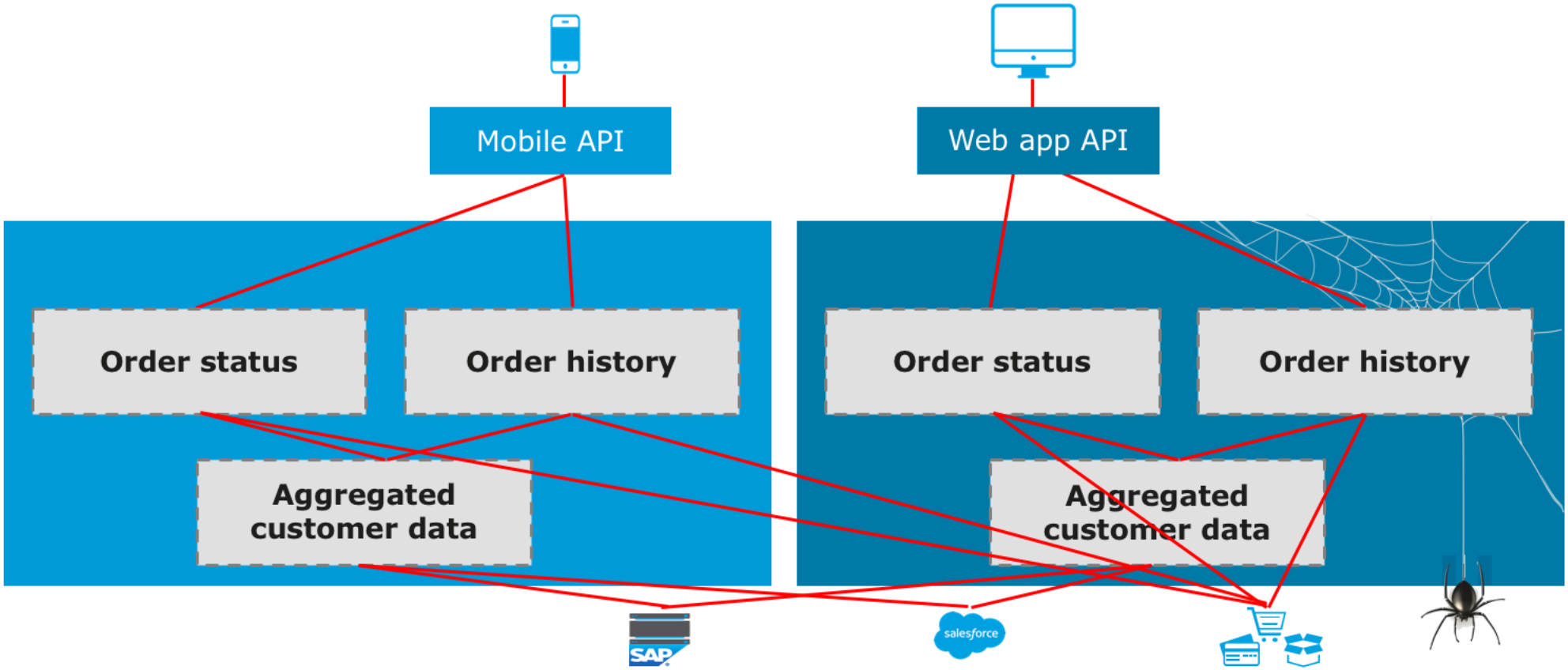


A common project-based approach

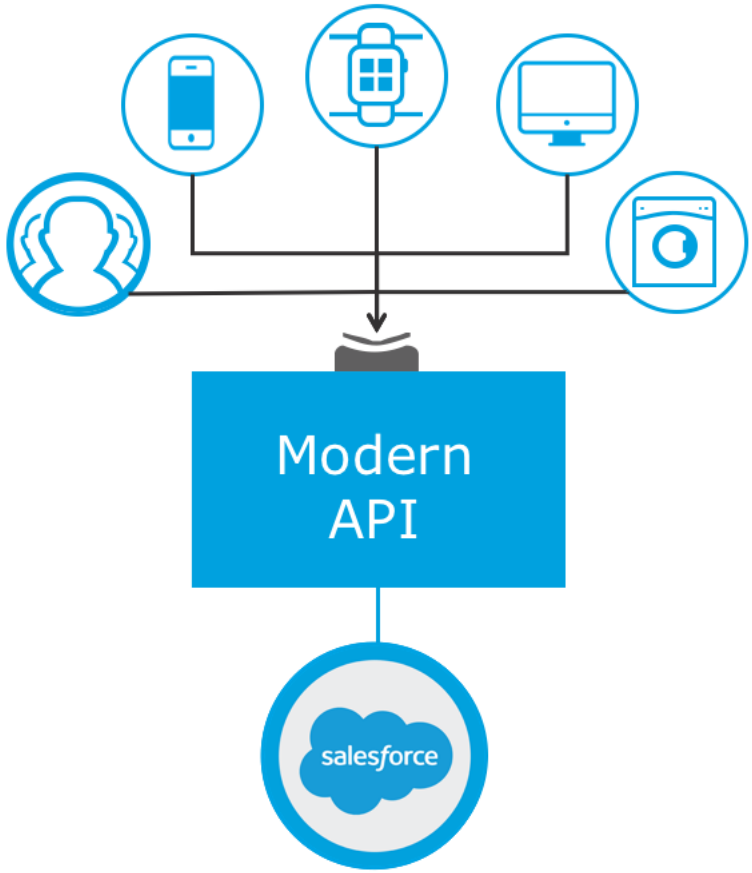
- On time and within budget
- Limited opportunity for reuse
- Tight coupling = brittleness
- Difficult to govern
- Meets business requirements



6 months later...

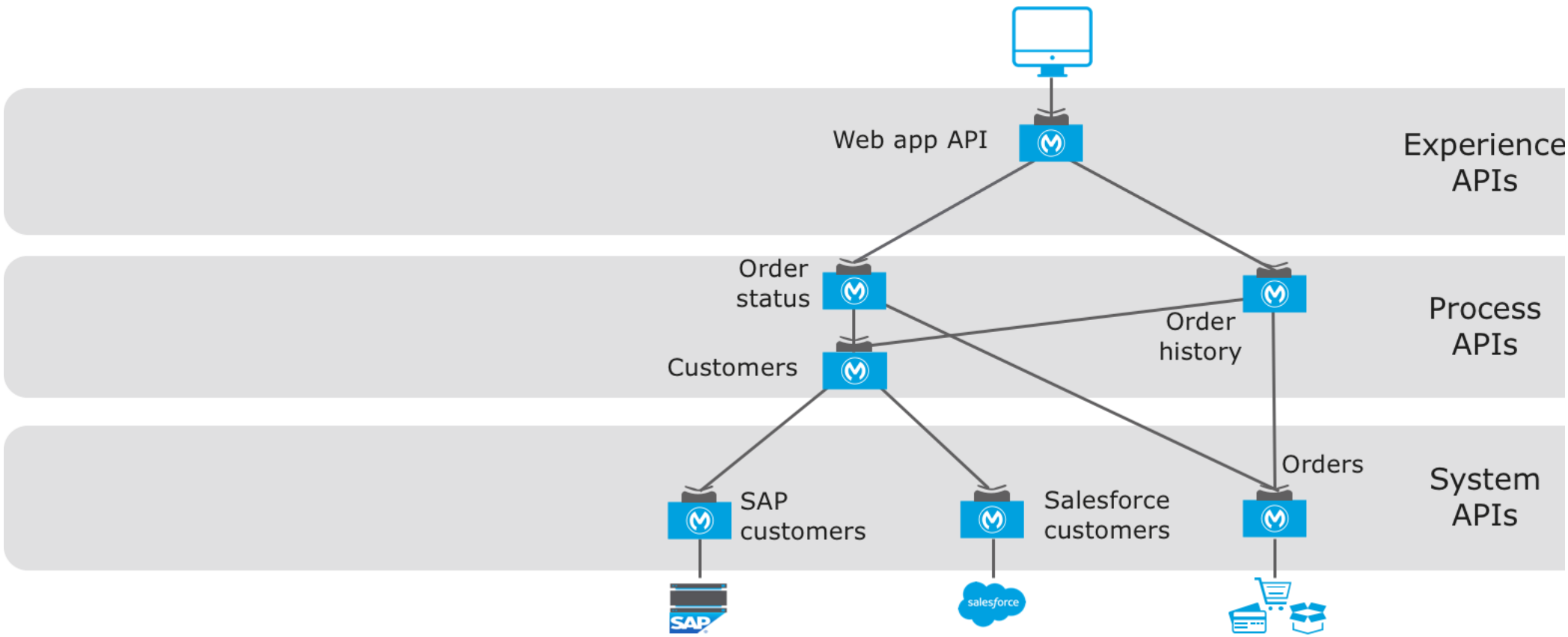


Modern API: The core enabler of a new operating model

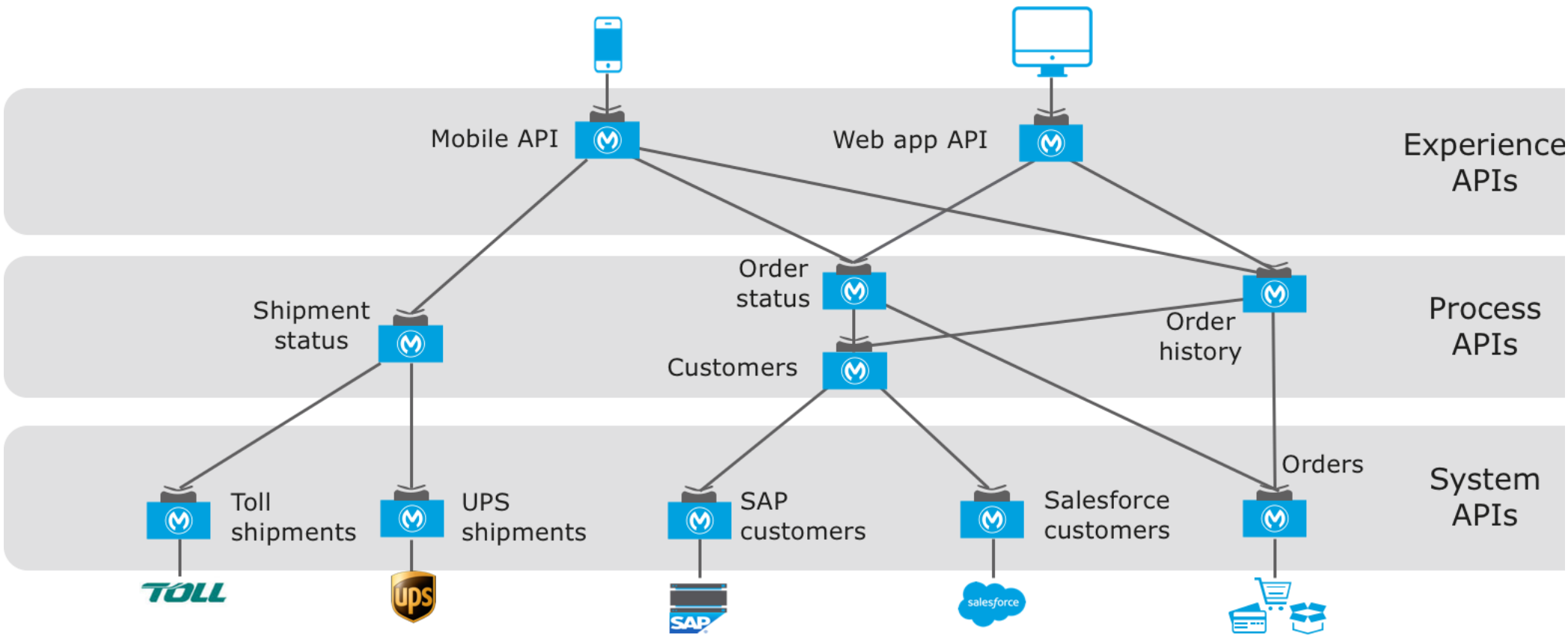


- Discoverable and accessible through self-service
- Productized and designed for ease of consumption
- Easily managed for security, scalability, and performance

The API-led connectivity approach



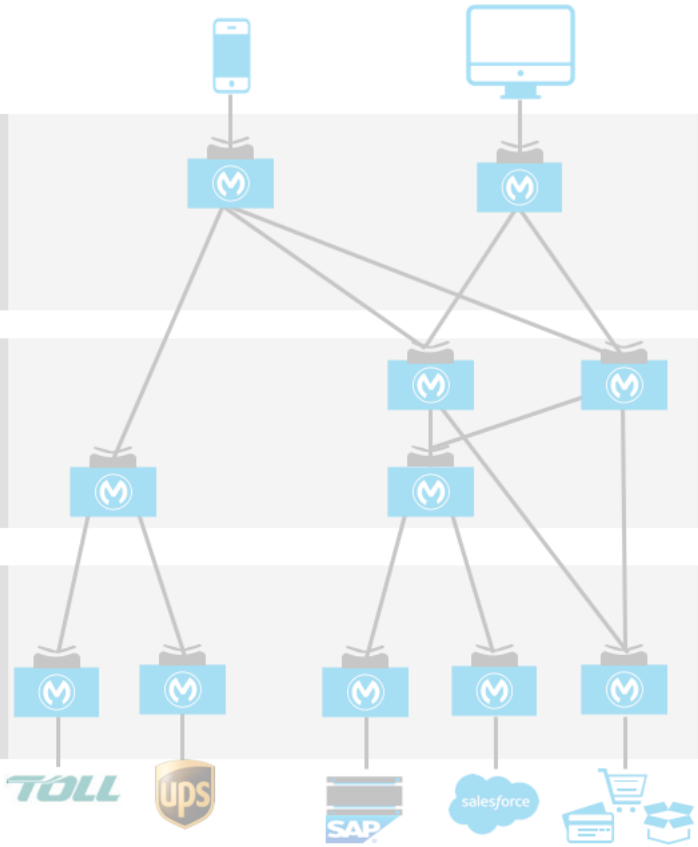
The API-led connectivity approach



Enable and empower the entire organization

MuleSoft

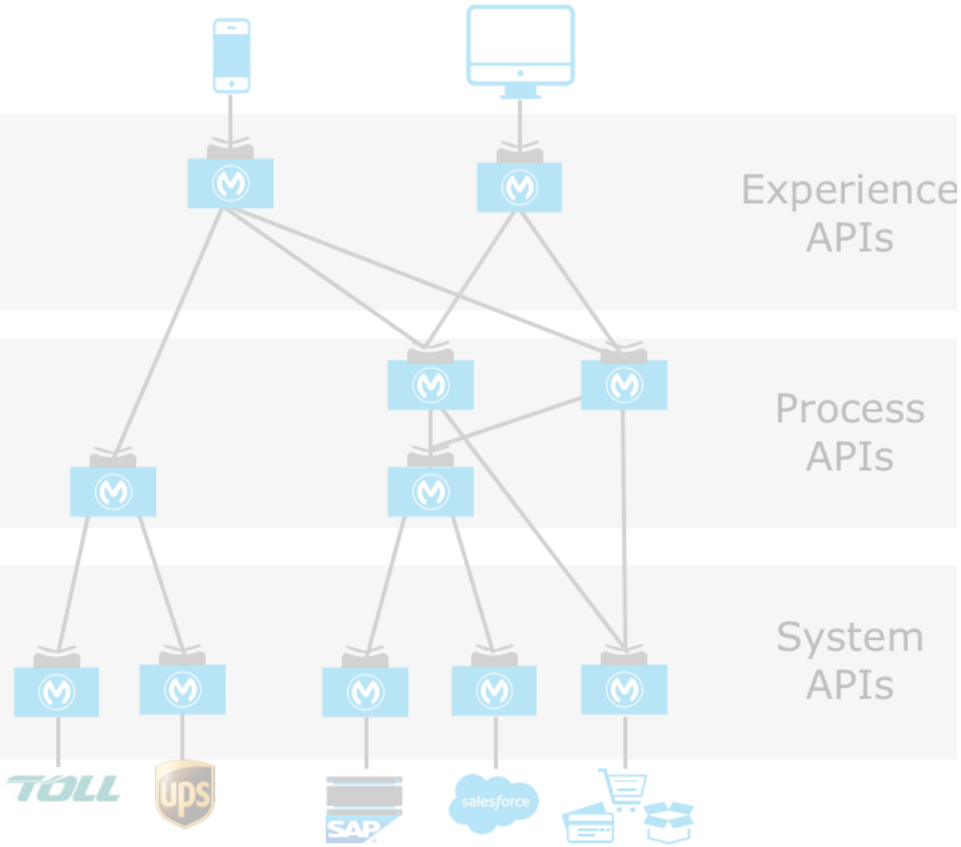
Developers	Discover, self-serve, reuse and **consume**
LOB IT	Discover, reuse assets and **compose** information
Central IT	**Unlock** assets and decentralize access



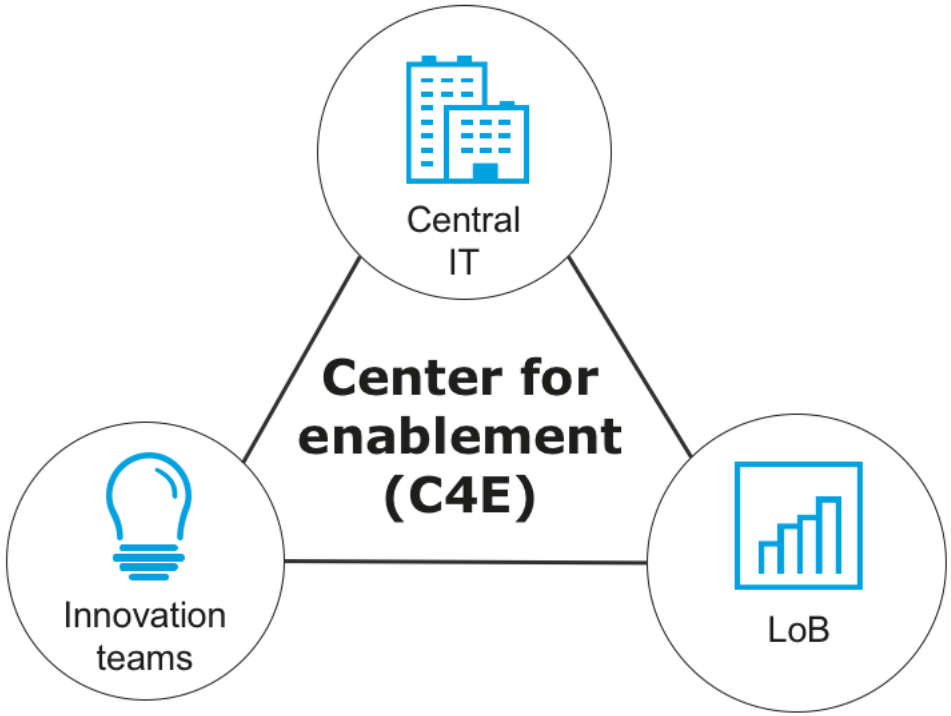
Drive outcomes with API-led connectivity

MuleSoft

- 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



C4E: Organizing differently to drive API-led connectivity



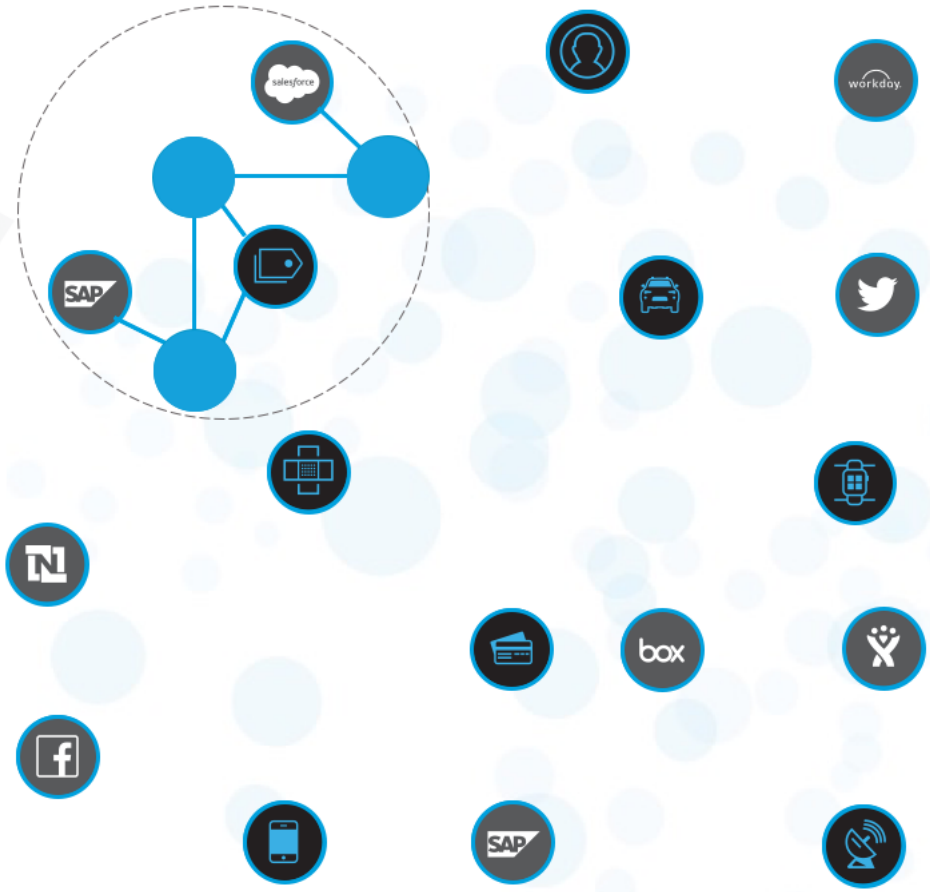
- C4E is a cross functional team
- C4E ensures that assets are
  - Productized and published
  - Consumable
  - Consumed broadly
  - Fully leveraged
- Success of C4E measured on asset consumption

Achieving an application network

Every project adds value to the application network

MuleSoft

Project 1 – API-led

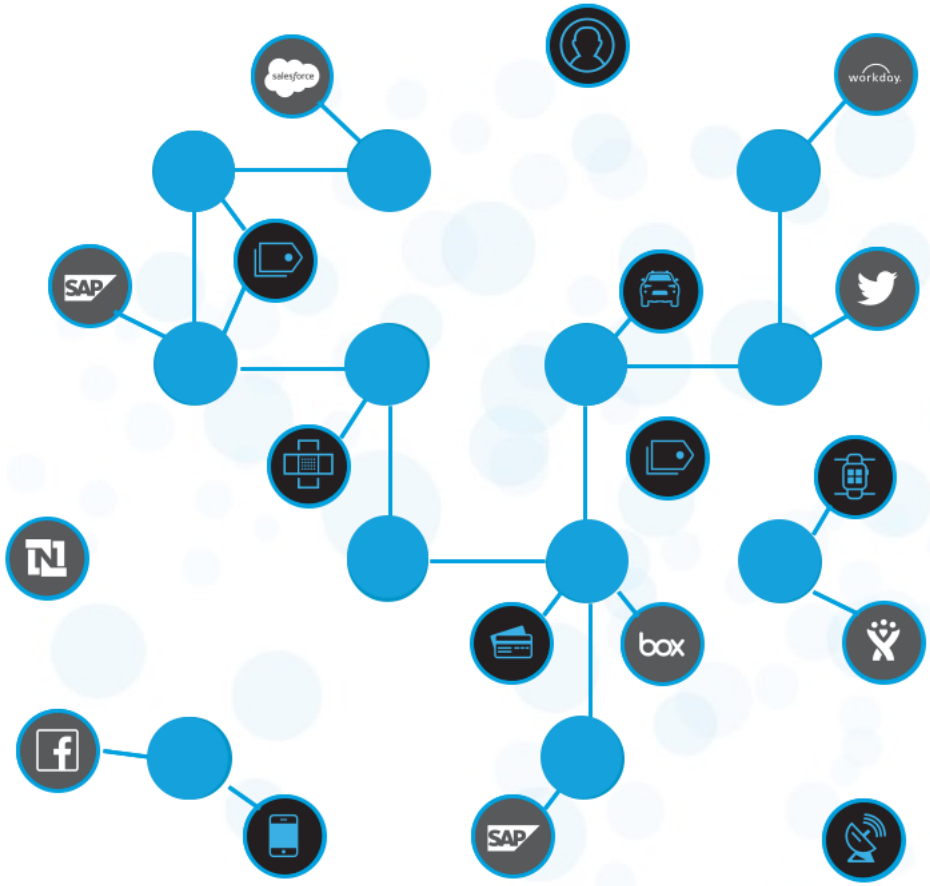


Application landscape

Every project adds value to the application network

MuleSoft

Project 1 – API-led



C4E

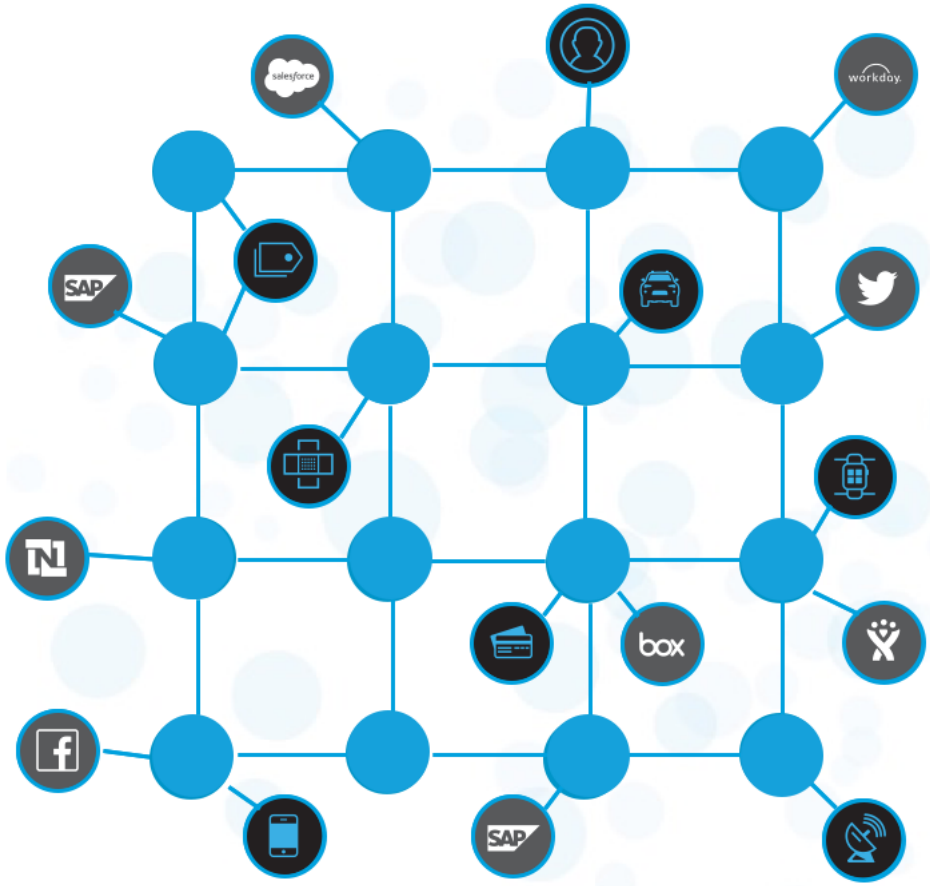
Self-serve assets on the application network

- Order status
- Shipments
- Orders
- Shipment status
- Customers

Every project adds value to the application network

MuleSoft

Project 1 – API-led

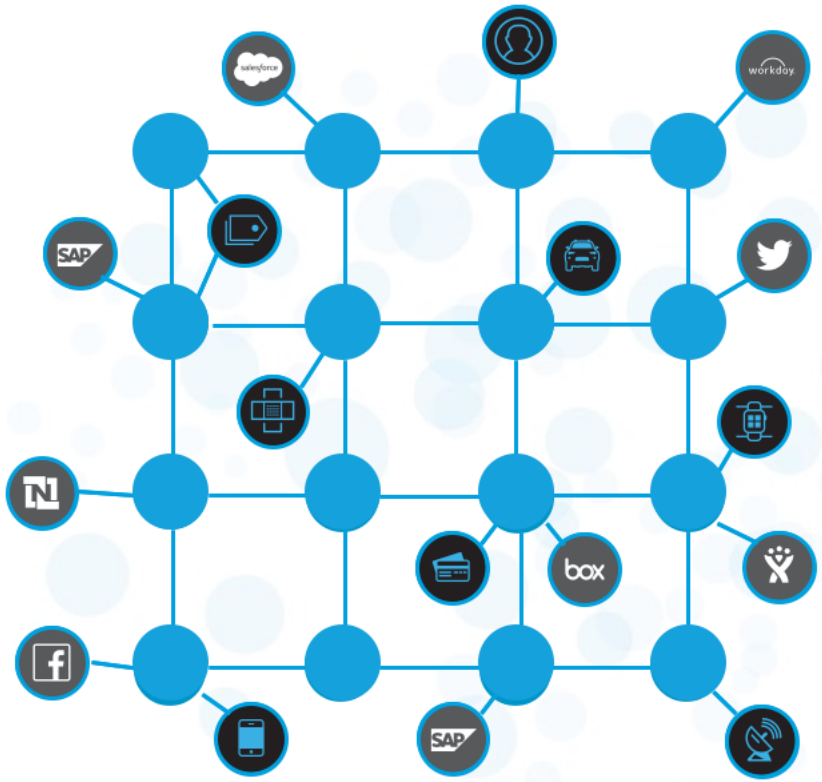


C4E

Self-serve assets on the application network

- Order status
- Shipments
- Orders
- Shipment status
- Customers

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**

Deconstructing APIs

What exactly is an API?

- An **API** is an **A**pplication **P**rogramming **I**nterface

- It provides the info 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 (API specification)**
   - 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

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

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

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
  - 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

Example RESTful web service calls

- Data and resources are represented using URIs
- Resources are accessed or changed using a fixed set of operations

- (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

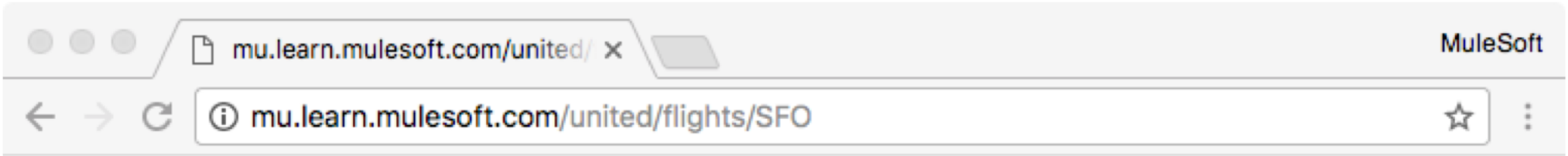
RESTful web service request methods

- **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

Example RESTful web service response

MuleSoft

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

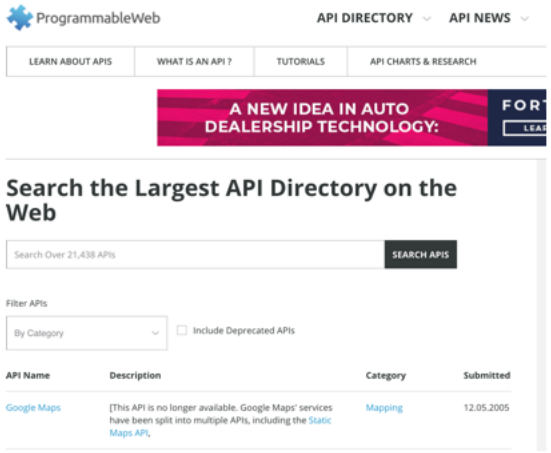
Learning about APIs

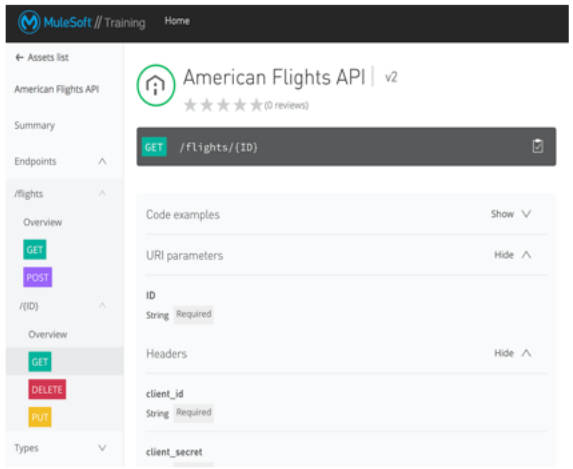
MuleSoft

- API documentation
  - Should include the list of all possible resources, how to get access to the API, and more
- API portals
  - Accelerate onboarding by providing developers a centralized place for discovering all the tools they need to successfully use the API, which could include
    - Documentation, tutorials, code snippets, and examples
    - A way to register applications to get access to the API
    - A way to provide feedback and make requests
    - A way to test the API by making calls to it
- Discover APIs in API directories and marketplaces
  - For example, **ProgrammableWeb**, which has over 19,000 APIs

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

- Browse the ProgrammableWeb API directory
- Explore the API reference for an API (like Twitter)
- Explore the API portal for an API to be used in the course



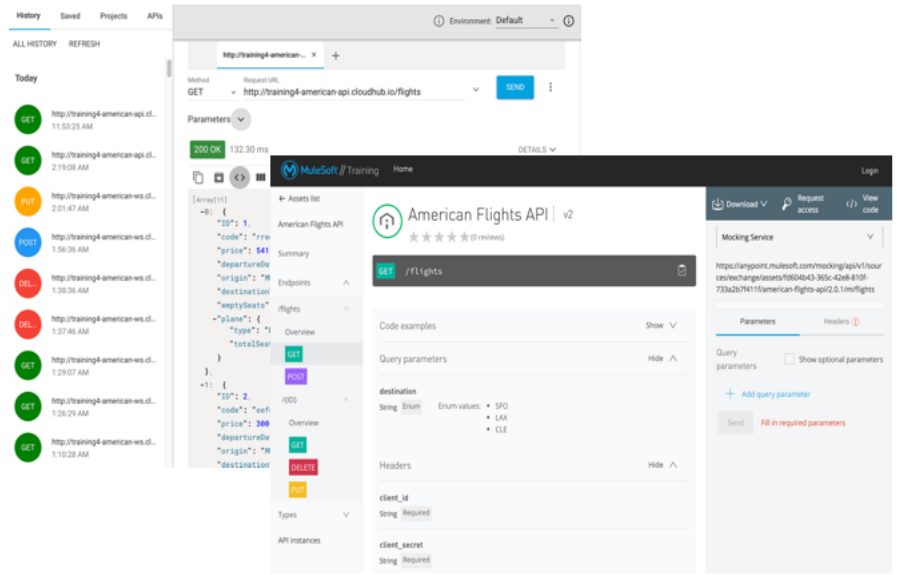


Calling RESTful web services

Calling RESTful web services

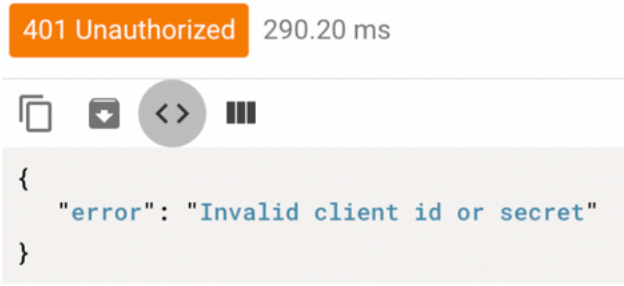
- To call web services, you need to write code or have a tool to make the HTTP requests
  - Need to be able to specify the HTTP method, request headers, and request body

- Example tools include
  - An API portal with an API console
  - Advanced Rest Client
  - Postman
  - A cURL command-line utility



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

Getting responses from web service calls

MuleSoft

- 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

```
post:
  body:
    application/json:
      type: AmericanFlight
      examples:
        input: !include examples/Amer
  responses:
    201:
      body:
        application/json:
          example:
            message: Flight added (bu
```

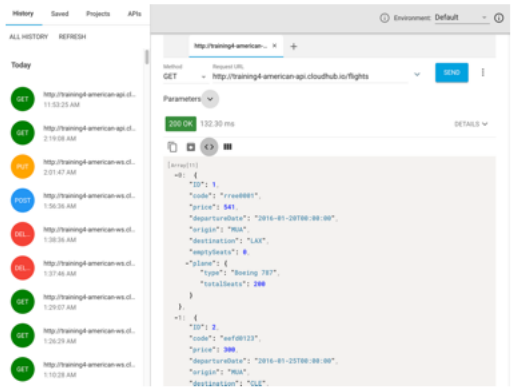
Common HTTP status codes

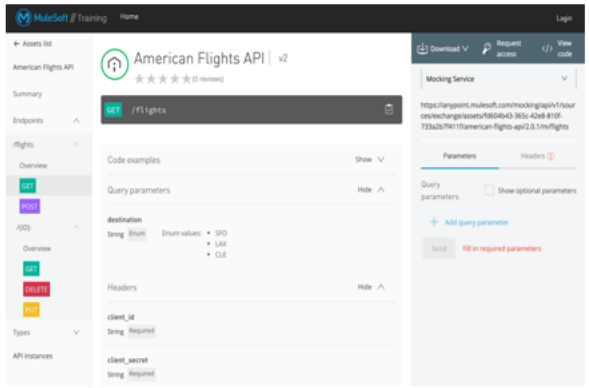
MuleSoft

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

Walkthrough 1-2: Make calls to an API

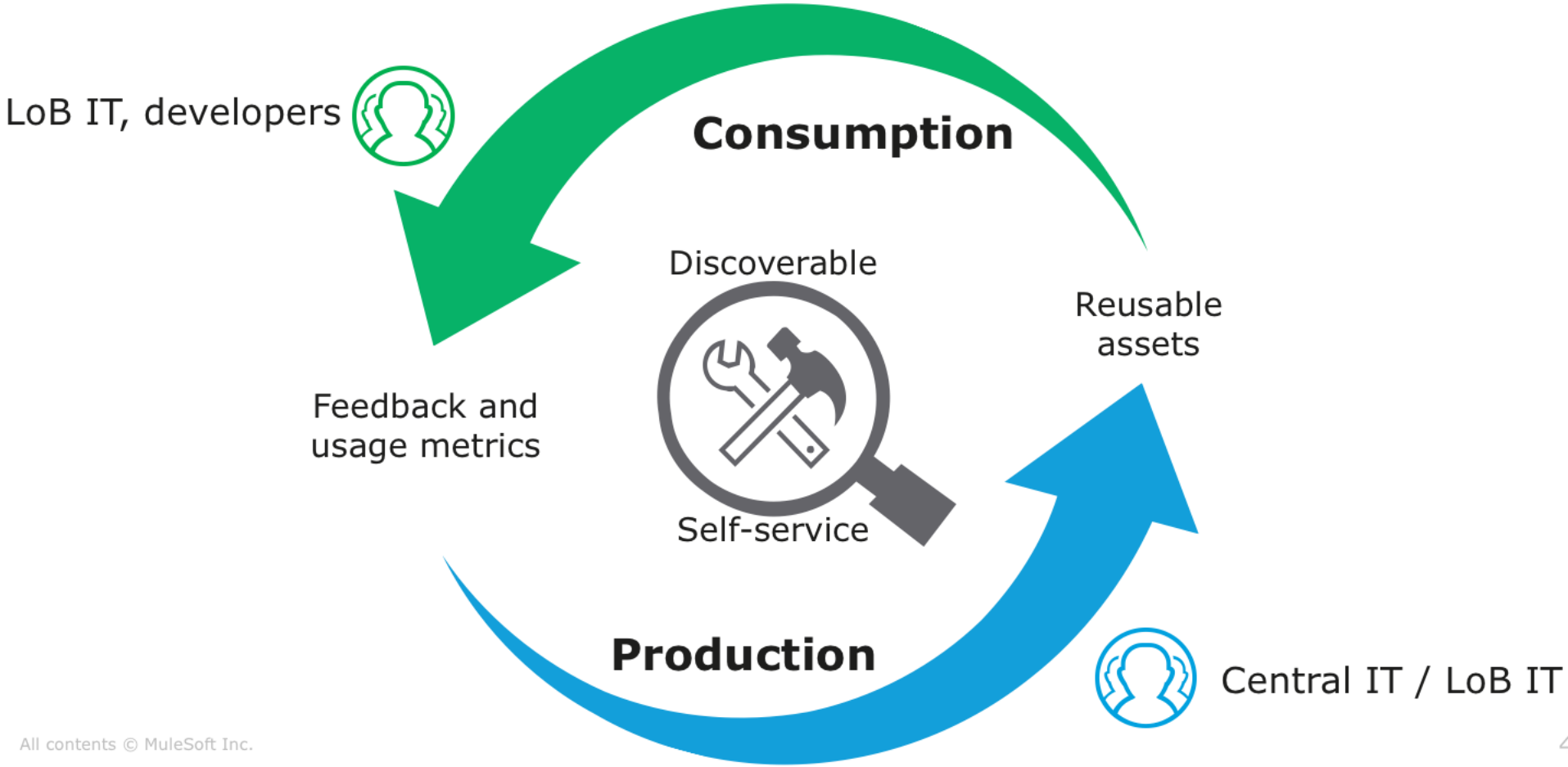
- Use ARC to make calls to an unsecured API (an implementation)
- Make GET, DELETE, POST, and PUT calls
- Use ARC to make calls to a secured API (an API proxy)
- Use the API console in an API portal to make calls to a managed API using a mocking service
- Use the API console to make calls to an API proxy endpoint





Successfully creating application networks using API-led connectivity

Producing discoverable and consumable assets is key



Designing for API success

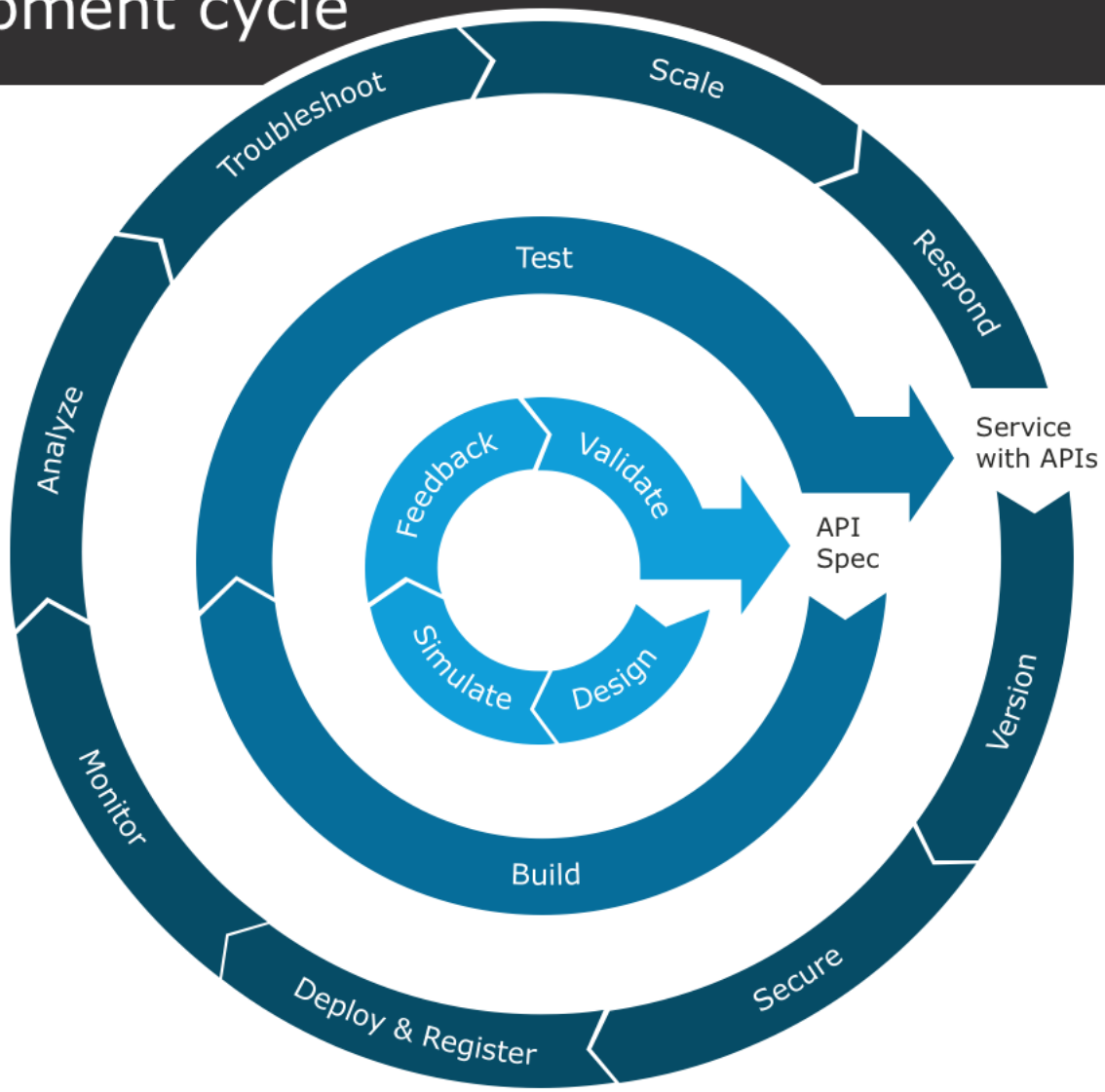
MuleSoft

- 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

API development cycle



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