Integration Pattern

Learning Program 2016

People matter, results count.

Objectives of Integration Pattern

- Purpose:
  - Understand Integration pattern for applications
- Product:
  - Integration Pattern
    - Study of Related Pattern
  - MuleSoft
- Process:
  - Understand various integration pattern and how to use in our applications.

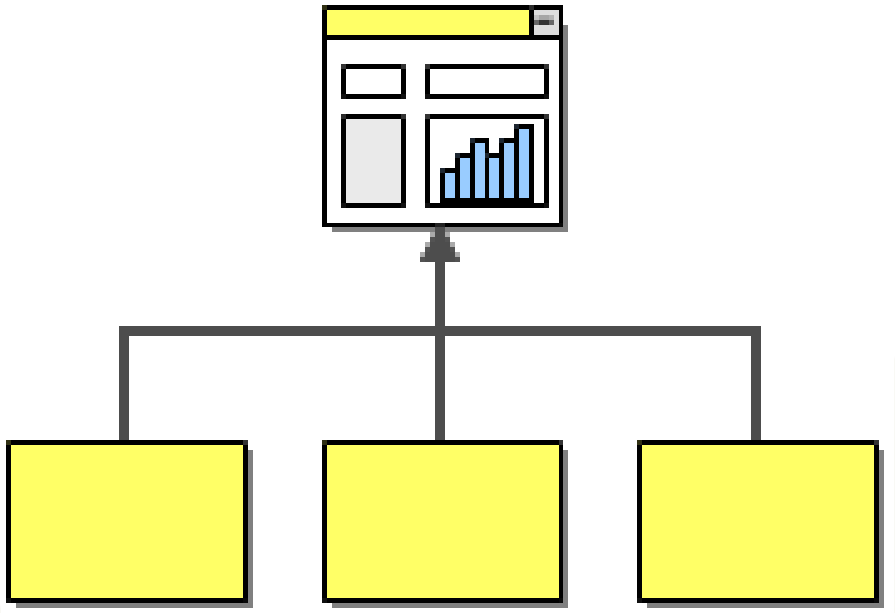
Table of Contents

Introduction

All integration solutions have to deal with a few fundamental challenges:

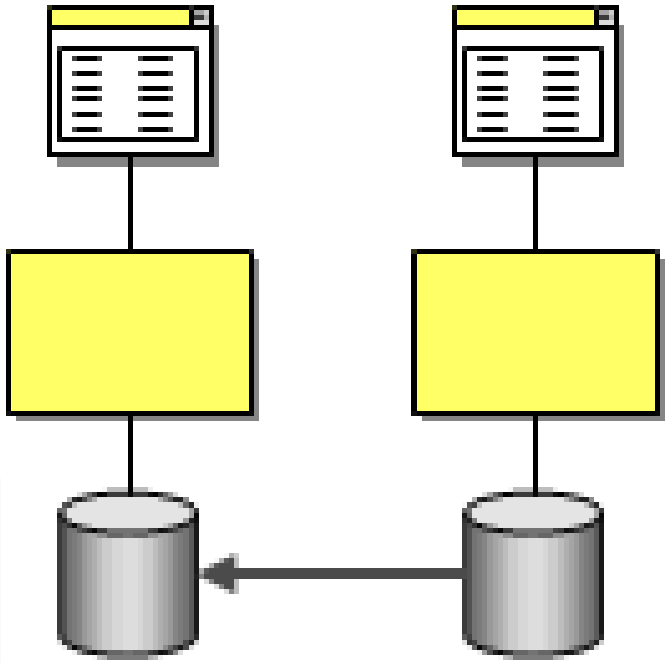
- Networks are unreliable
- Networks are slow
- Any two applications are different
- Change is inevitable

Information Portal



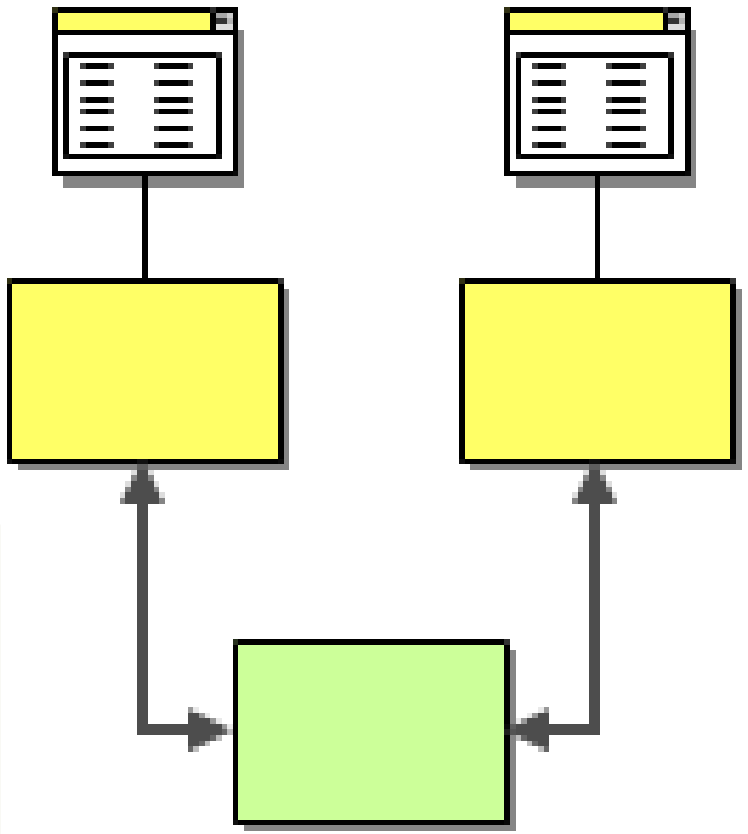
Information portals aggregate information from multiple sources into a single display to avoid having the user access multiple systems for information.

Data Replication



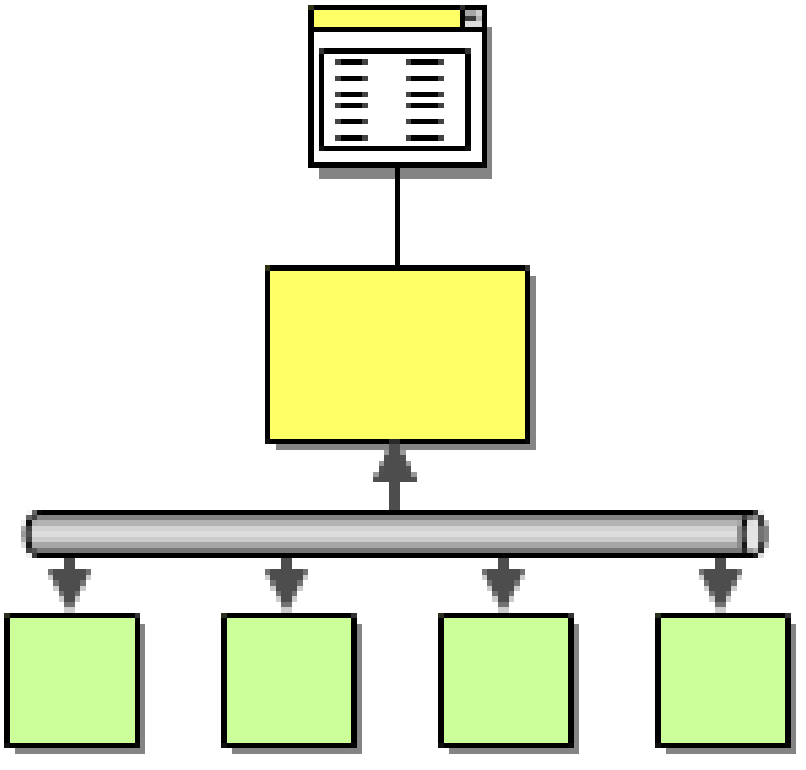
Many business systems require access to the same data.

Shared Business Function



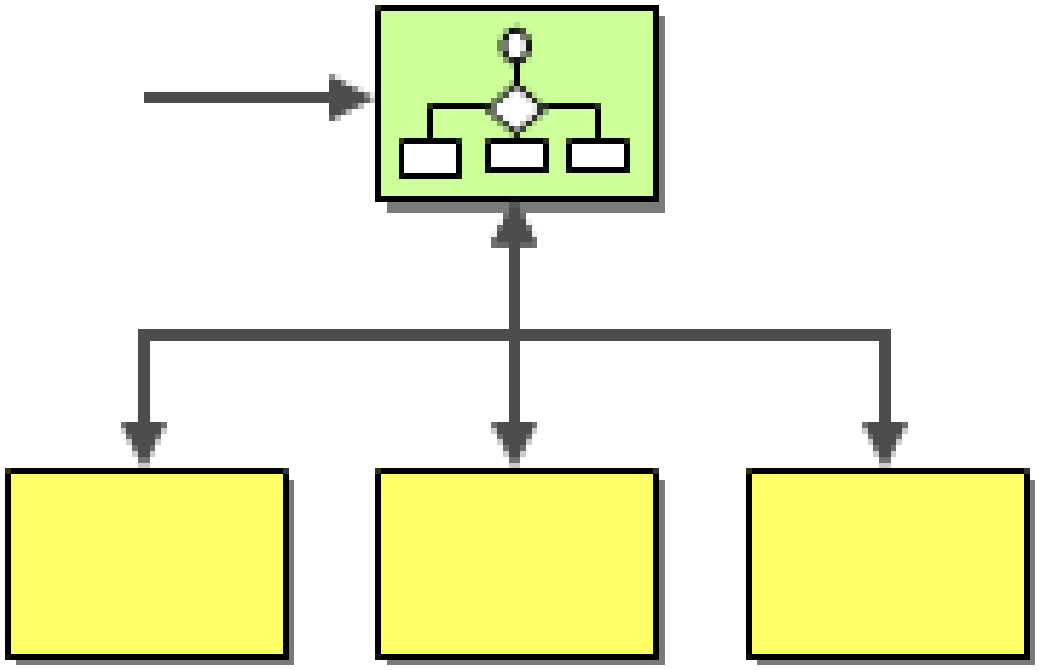
Many business applications store redundant data, they also tend to implement redundant functionality

Service Oriented Architecture



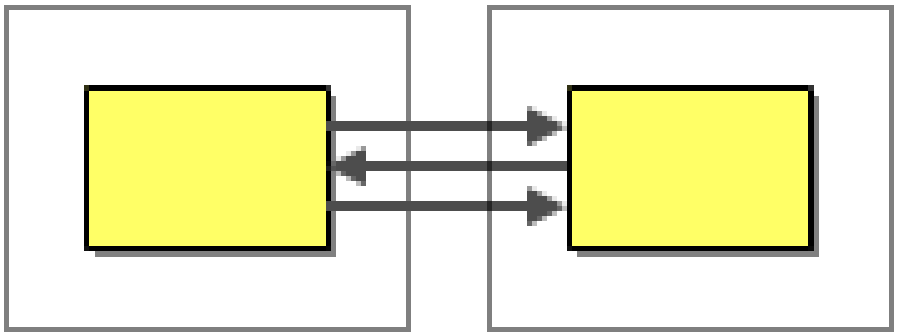
Shared business functions are often referred to as services. A service is a well-defined function that is universally available and responds to requests from “service consumers”.

Distributed Business Process



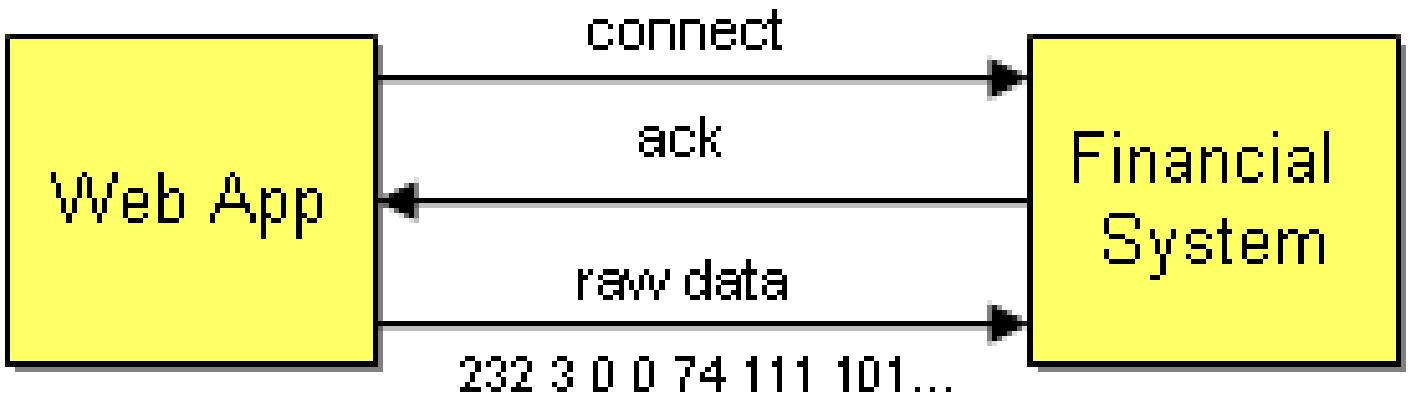
The boundaries between a service-oriented architecture and a distributed business can blur. For example, you could expose all relevant business functions as service and then encode the business process inside an application that accesses all services via an SOA.

Business to Business



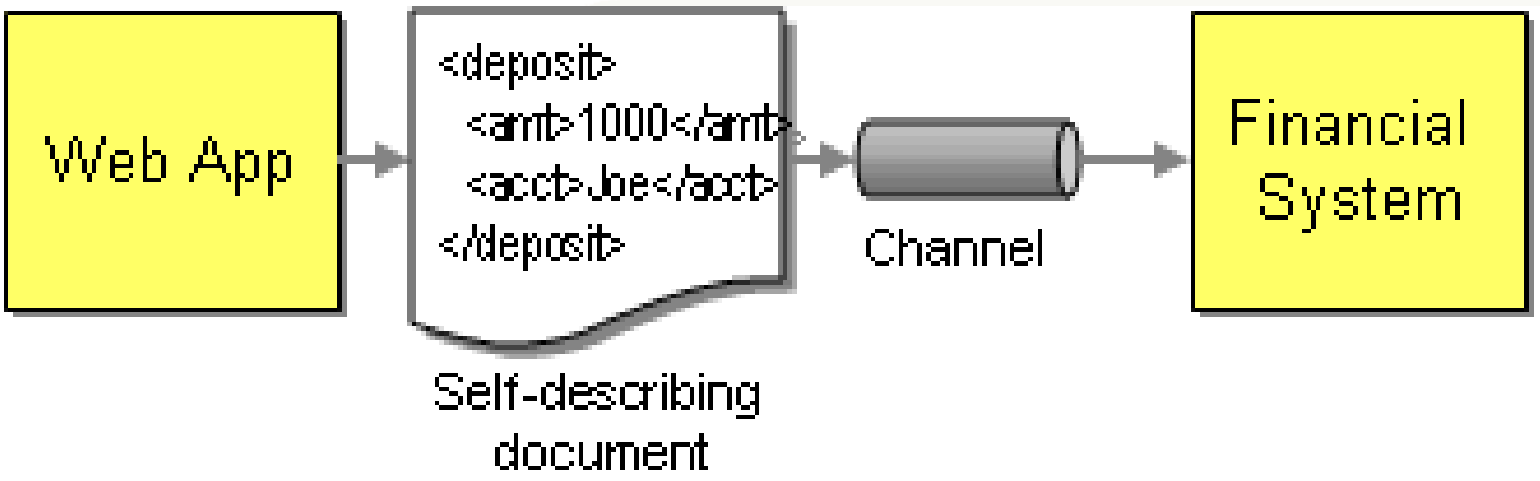
Business functions may be available from outside suppliers or business partners

Tightly Coupled System

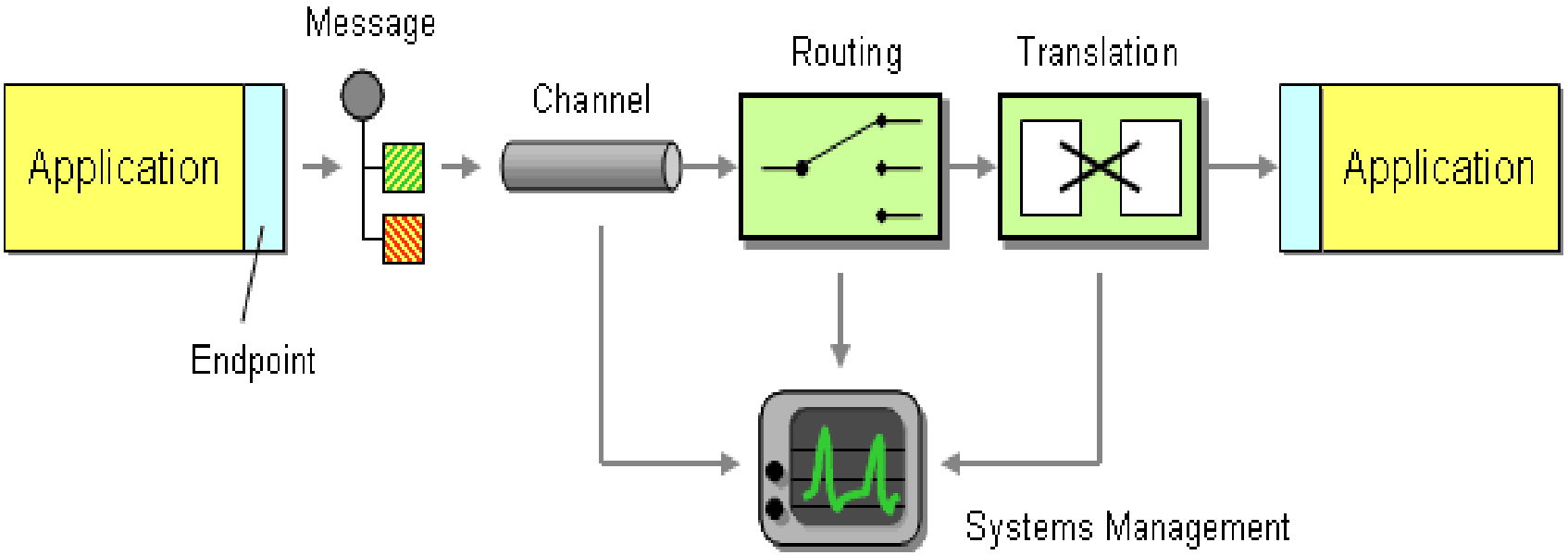


Loose Coupling

- The core principle behind loose coupling is to reduce the assumptions two parties (components, applications, services, programs, users) make about each other when they exchange information.

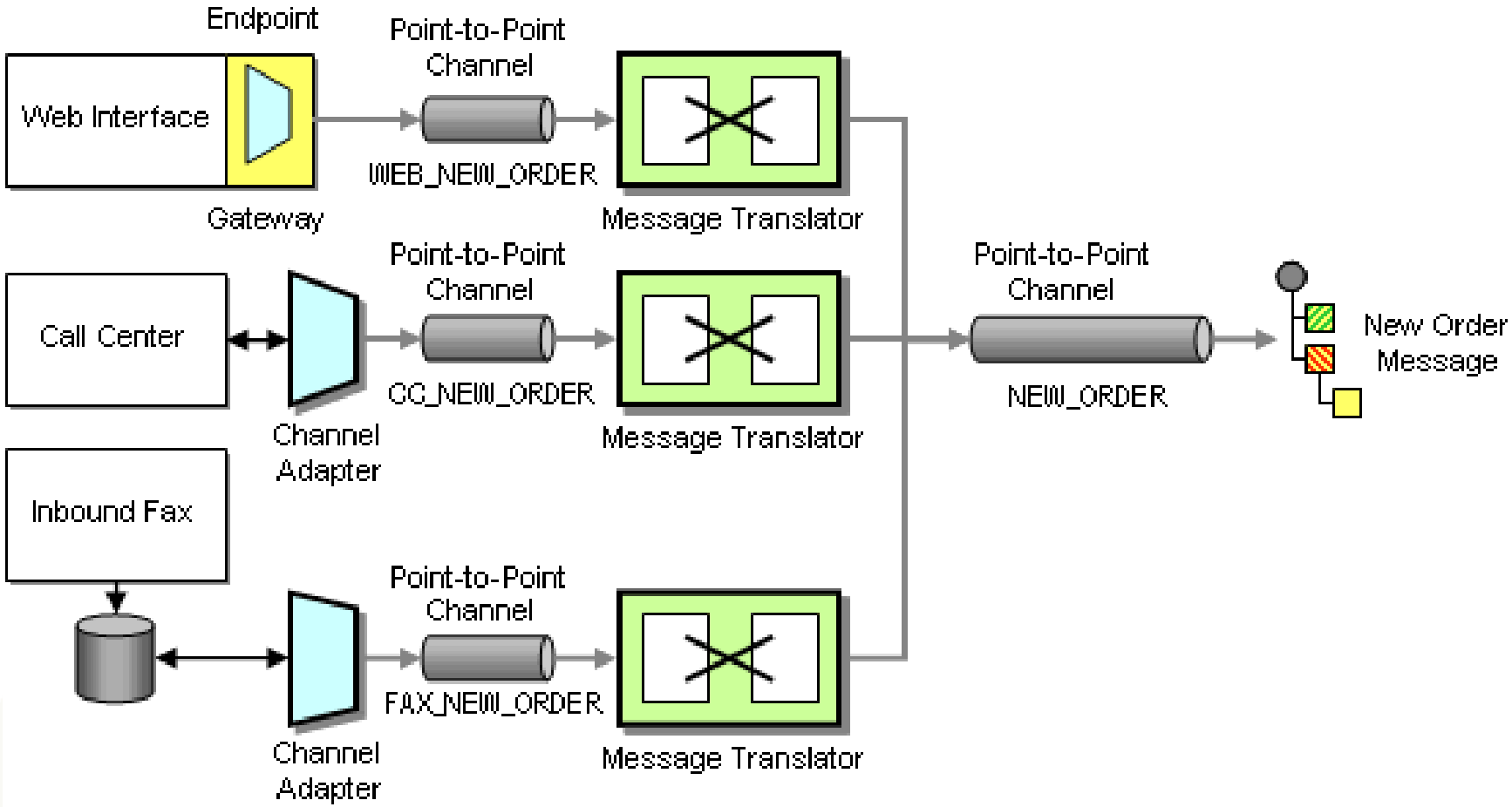


Basic Elements of an Integration Solution

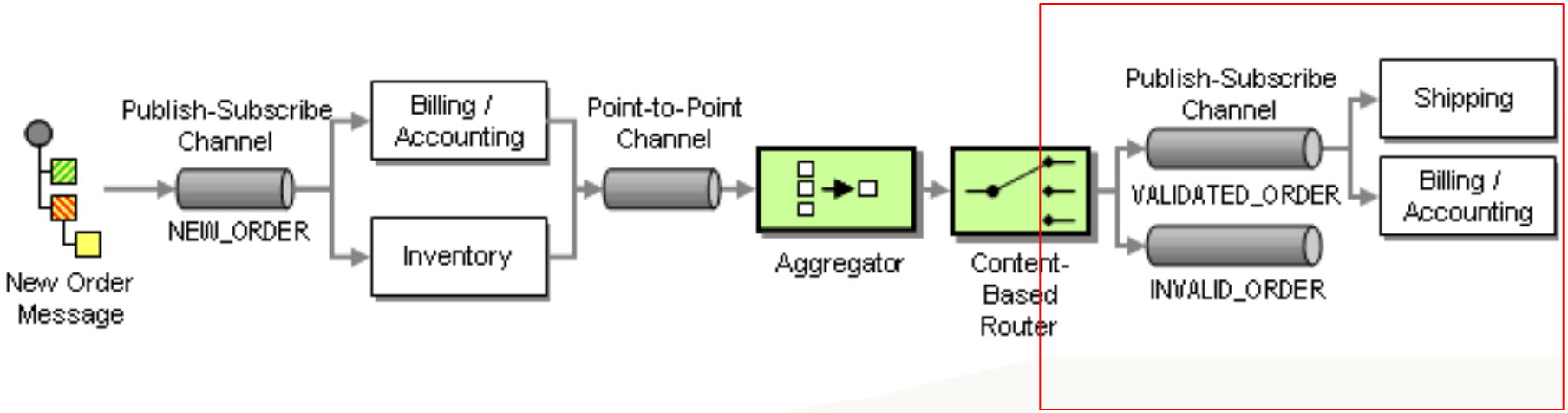


Channel
Message
Translation
Routing – Message Broker
Endpoint – Special Piece of Code / Channel Adapter -> by Vendor

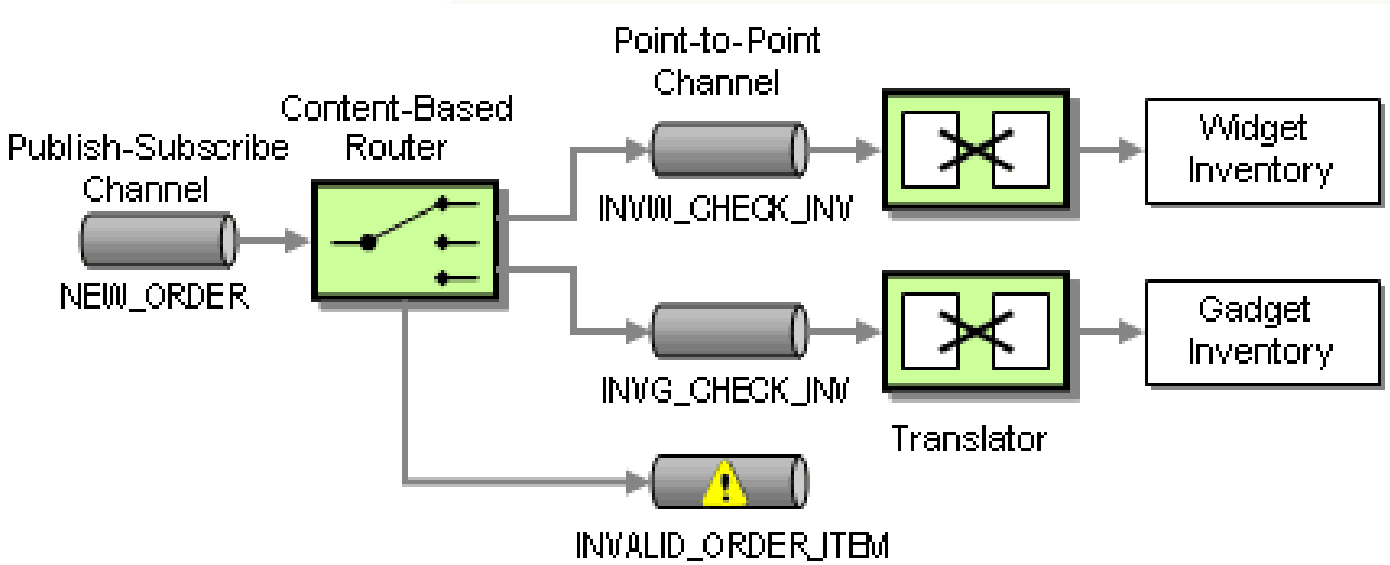
Taking Orders From Three Different Channels



Order Processing Implementation using Asynchronous Messaging



Routing the Inventory Request



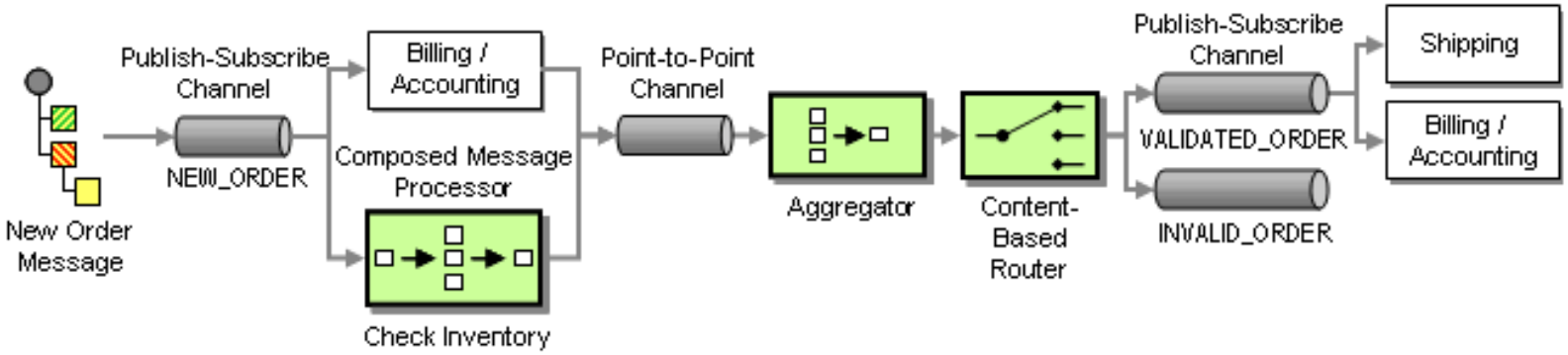
Terms

- ***Splitter**, a component that breaks a single message into multiple individual* messages.
- ***Aggregator** ,* the component that can combine multiple messages into a single message.
- A *Content **Enricher** is a* component that adds missing data items to an incoming message.
- The combination of a *Splitter, a Router and an Aggregator is fairly common. We refer to it as a **Composed Message Processor.***
- ***Process Manager** component that receives a New Order message (which includes the current* shipping and billing address0 and publishes two separate messages to the billing (or shipping) system

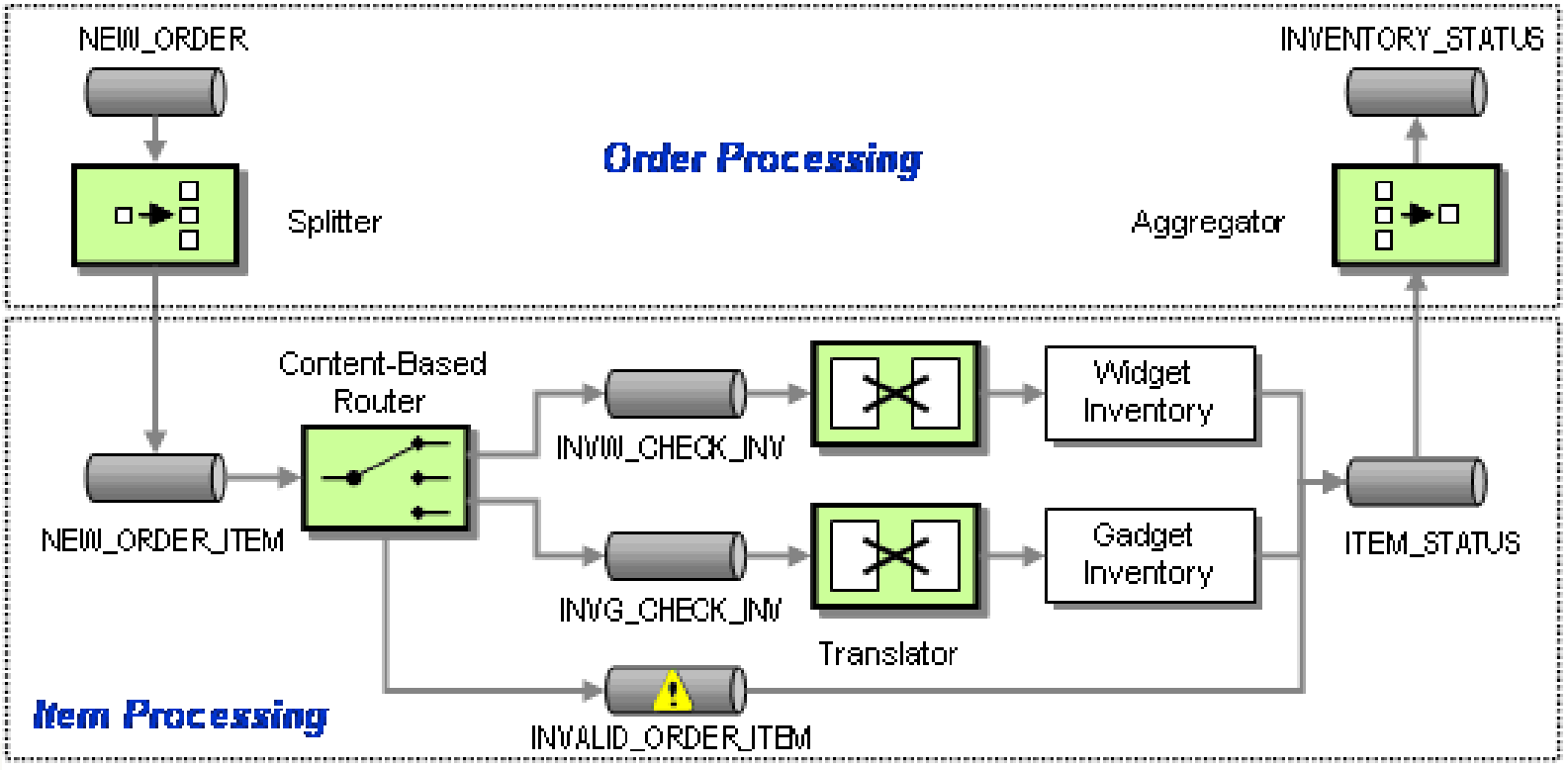
Terms

- A ***Dynamic Recipient List*** *is the* combination of two Message Routing patterns.
- ***Message Store*** *can provide us with some important business metrics such as the average time to* fulfill an order.

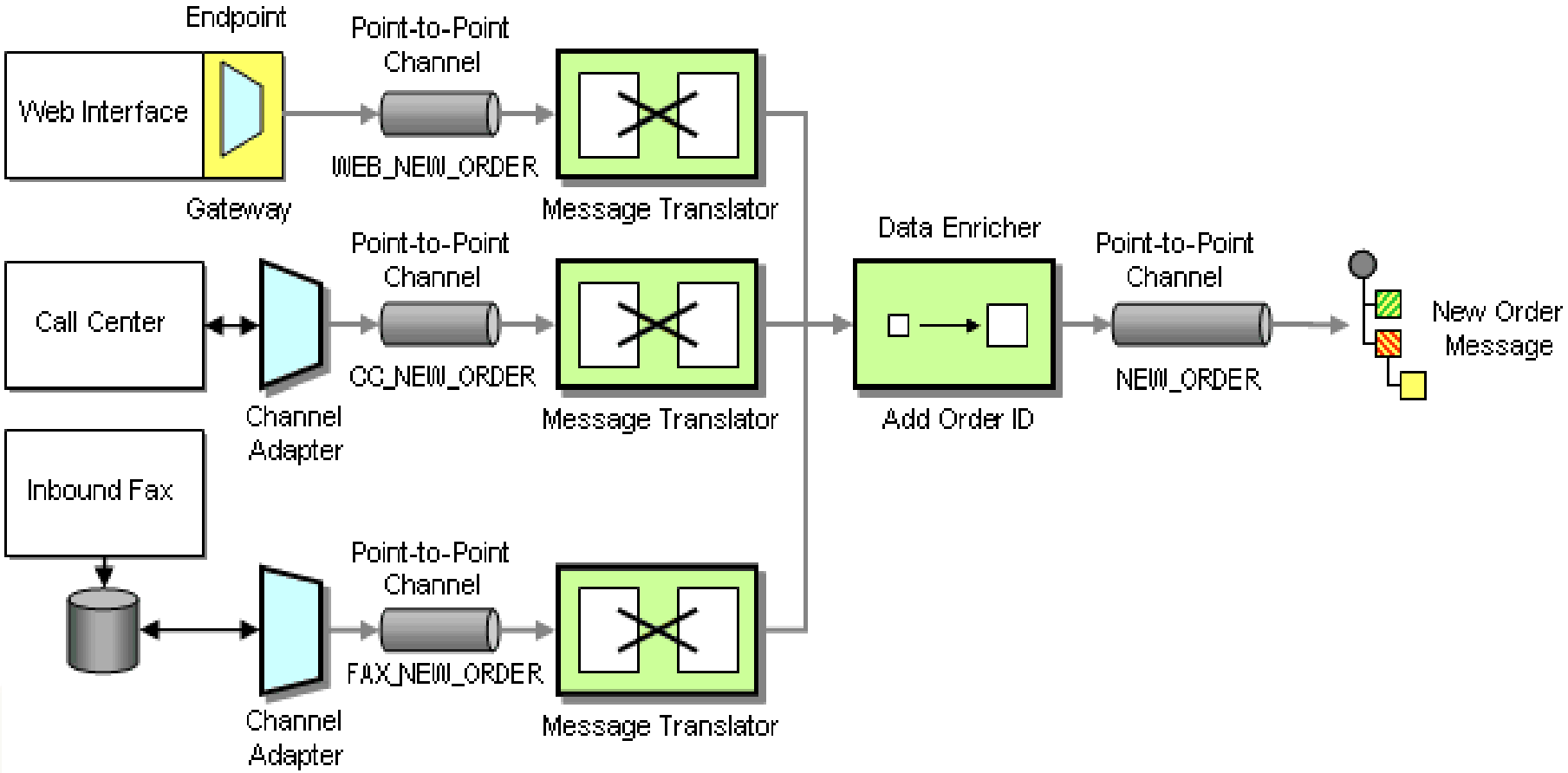
Revised Order Process Implementation



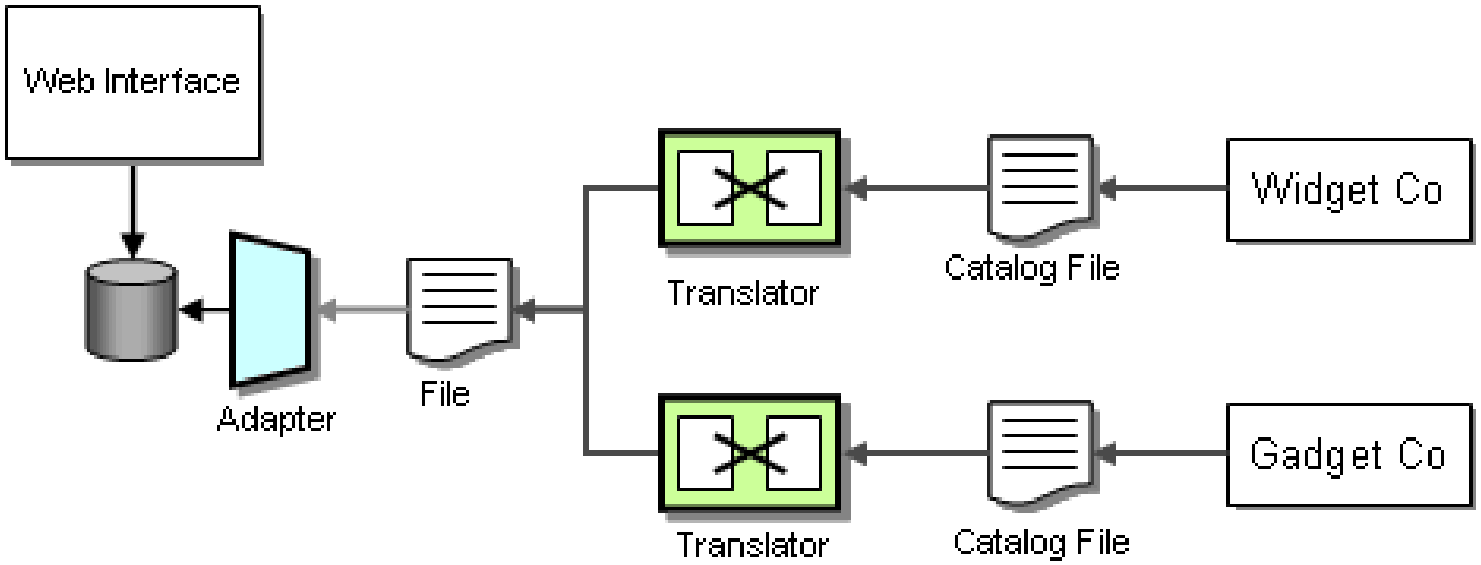
Processing Order Items Individually



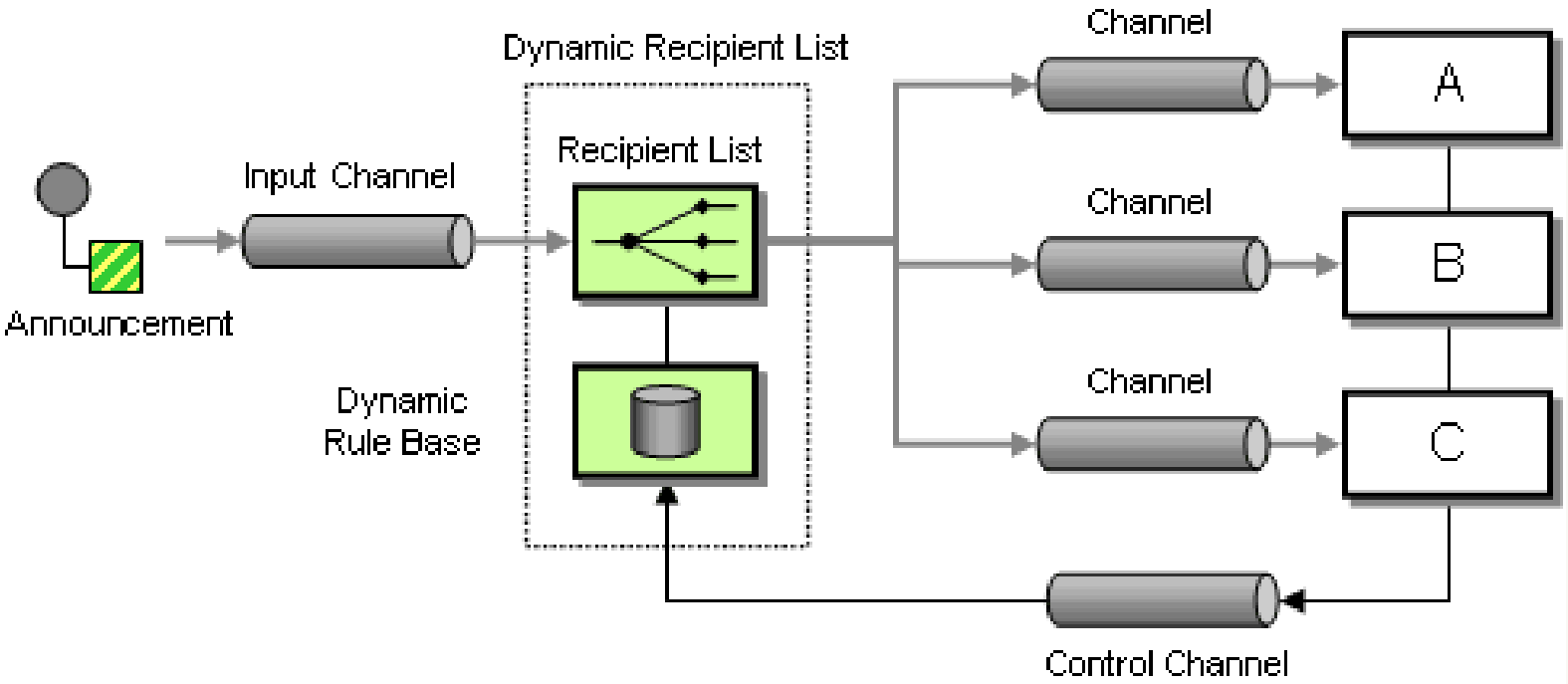
Taking Orders With Enricher



Updating Catalog Data via File Transfer



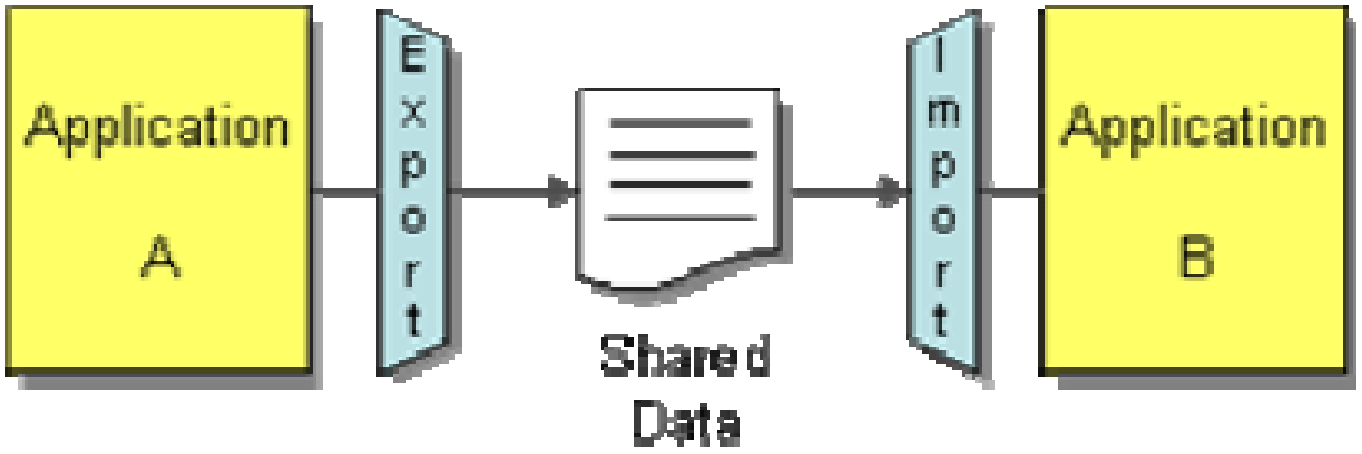
Sending Announcements With a Dynamic Recipient List



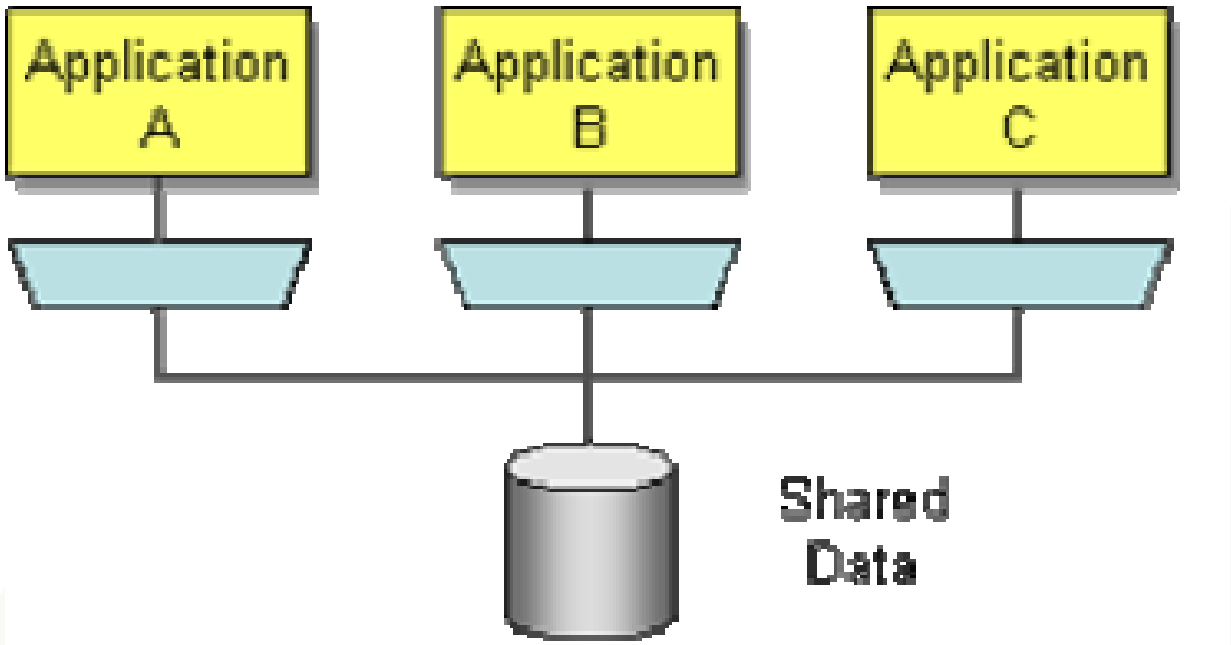
Integration Styles

- File Transfer
- Shared Database
- Remote Procedure Invocation
- Messaging

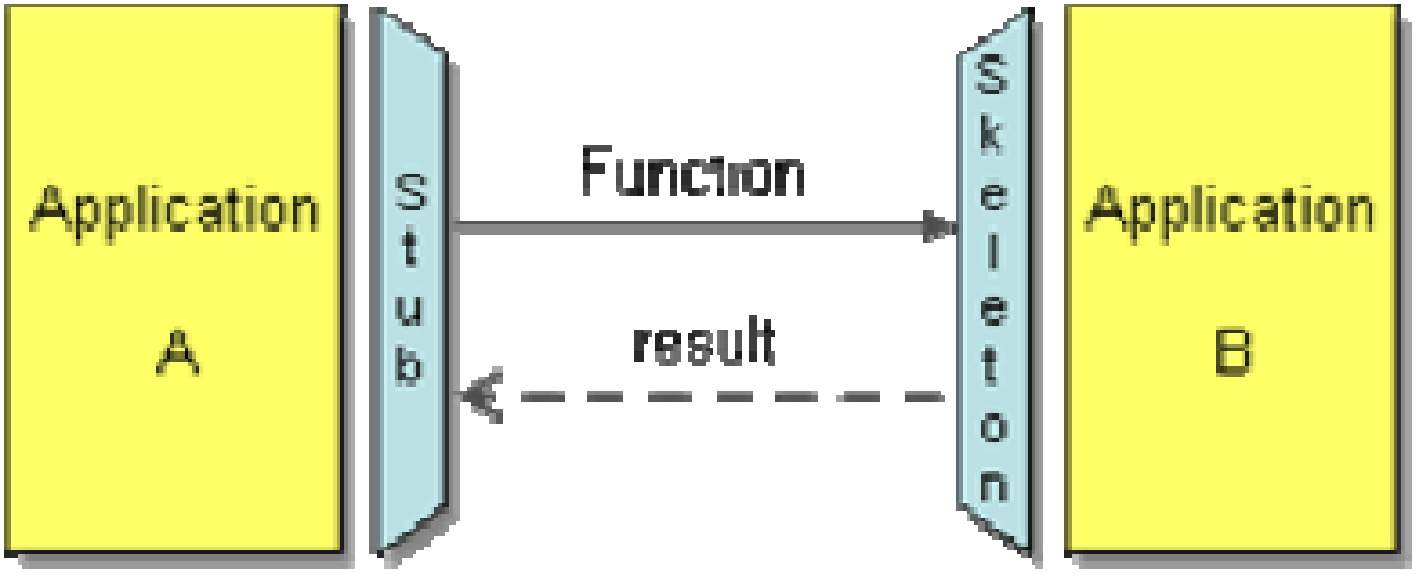
File Transfer



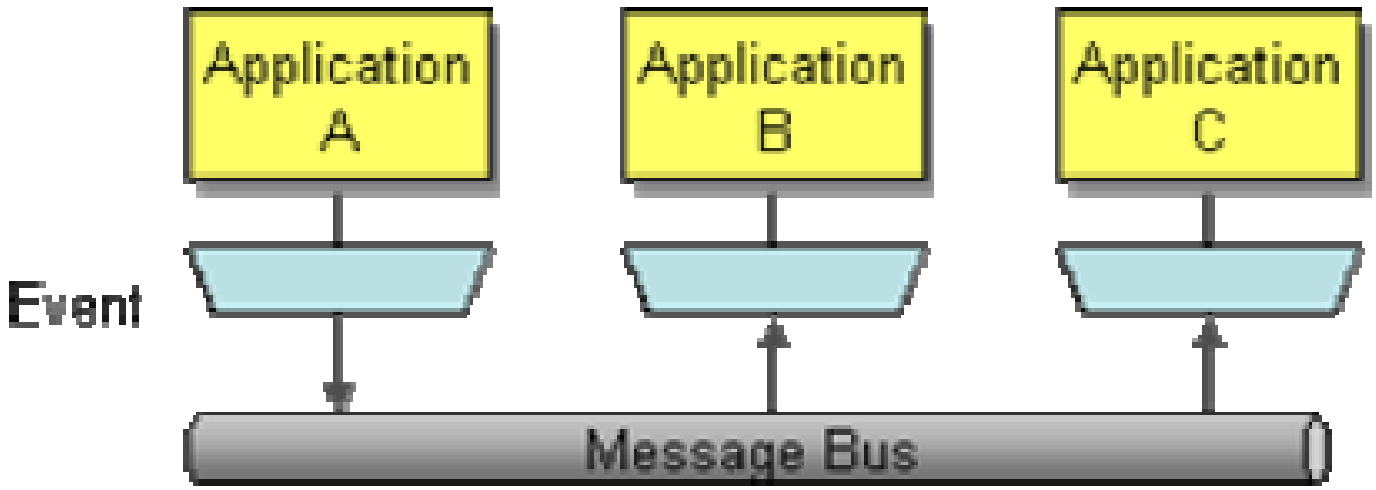
Shared Database



Remote Procedure Invocation



Messaging



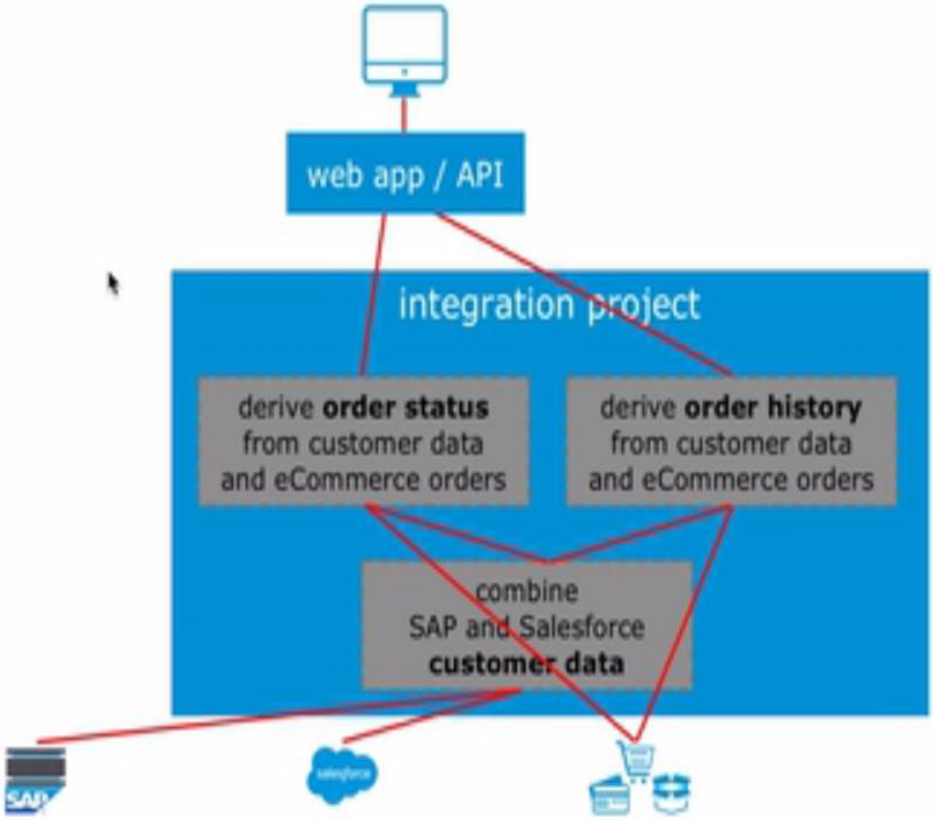
Message Channel – Connects Sender and Receiver
Message Router - where to address the data
Message Translator - that will convert the data to the receiver's format

An application that wishes to use messaging will implement ***Message Endpoints*** *to perform the actual sending and* receiving.

A traditional project-based approach

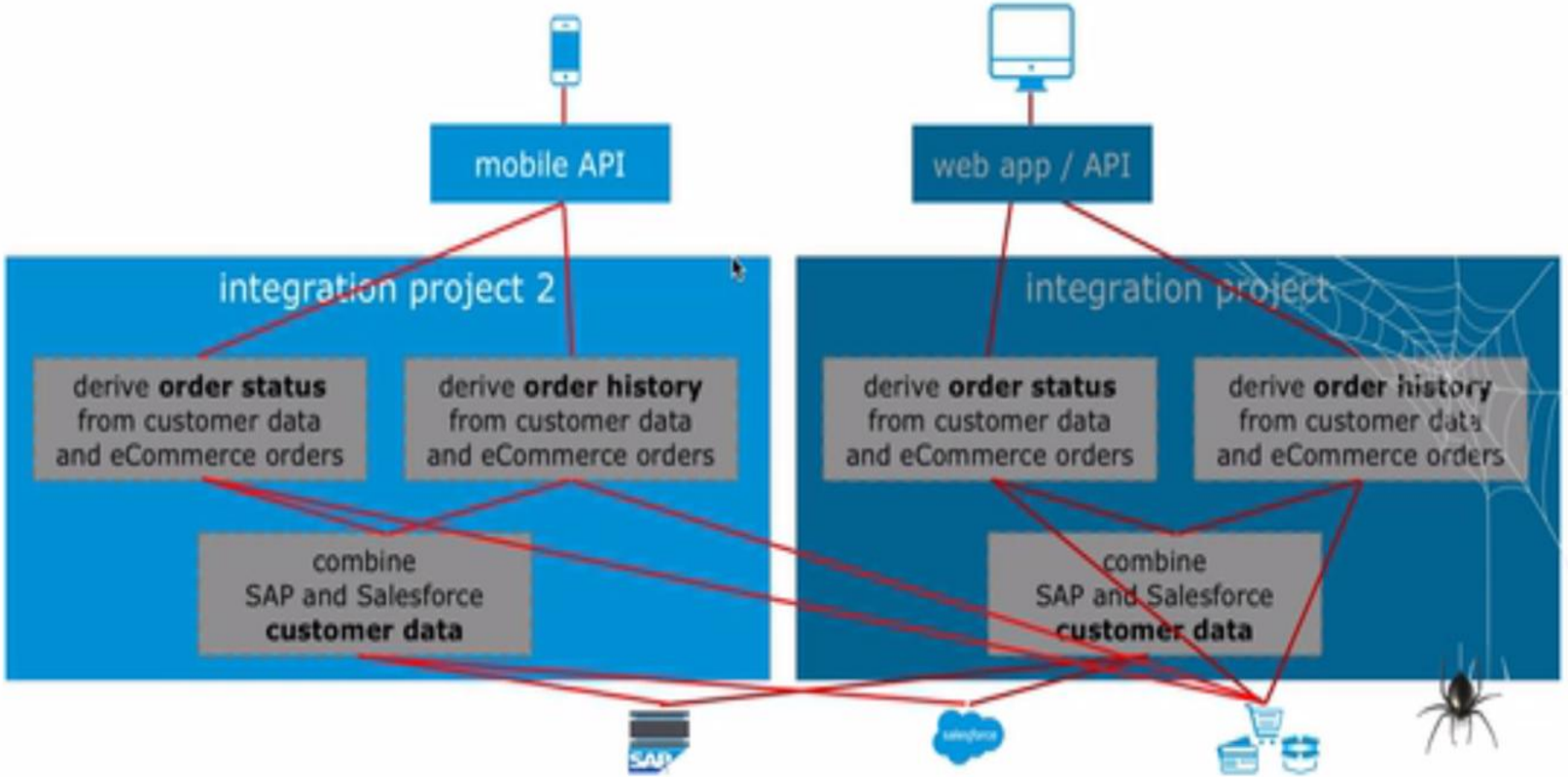
MuleSoft

- On time
- Within budget
- Meets requirements
- No reuse
- Tightly coupled to apps
- Lack of governance



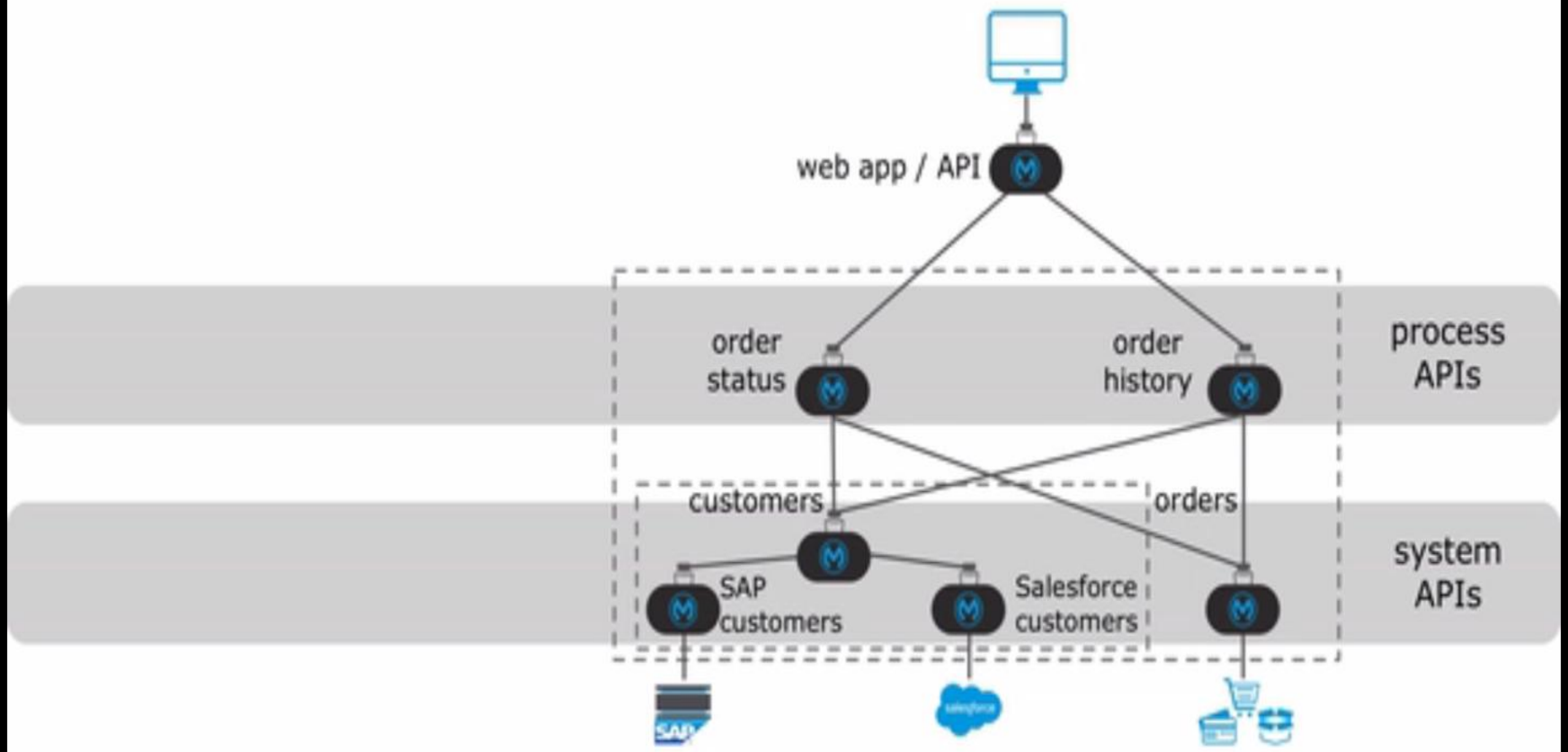
6 months later...

MuleSoft

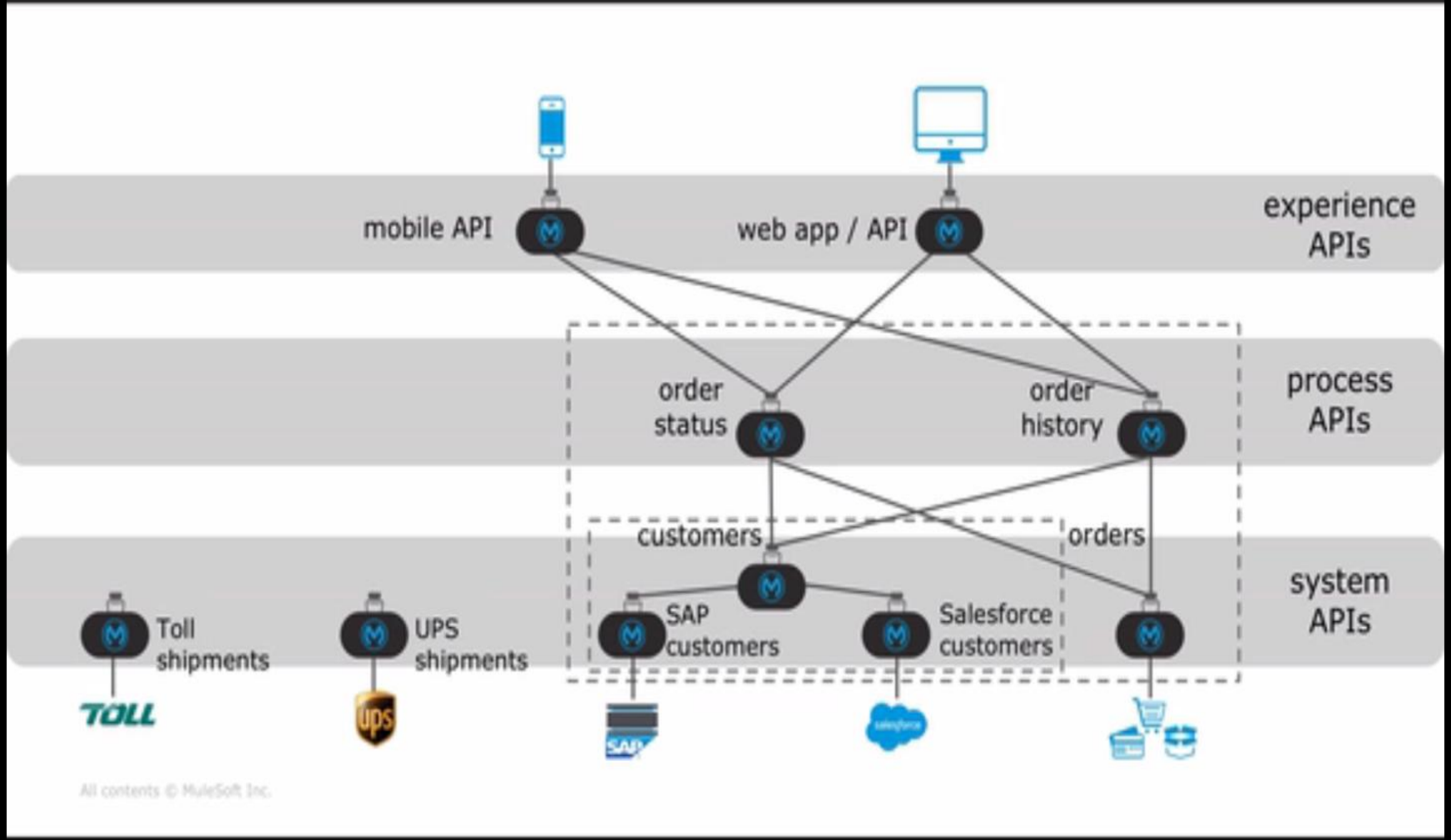


The API-led connectivity approach

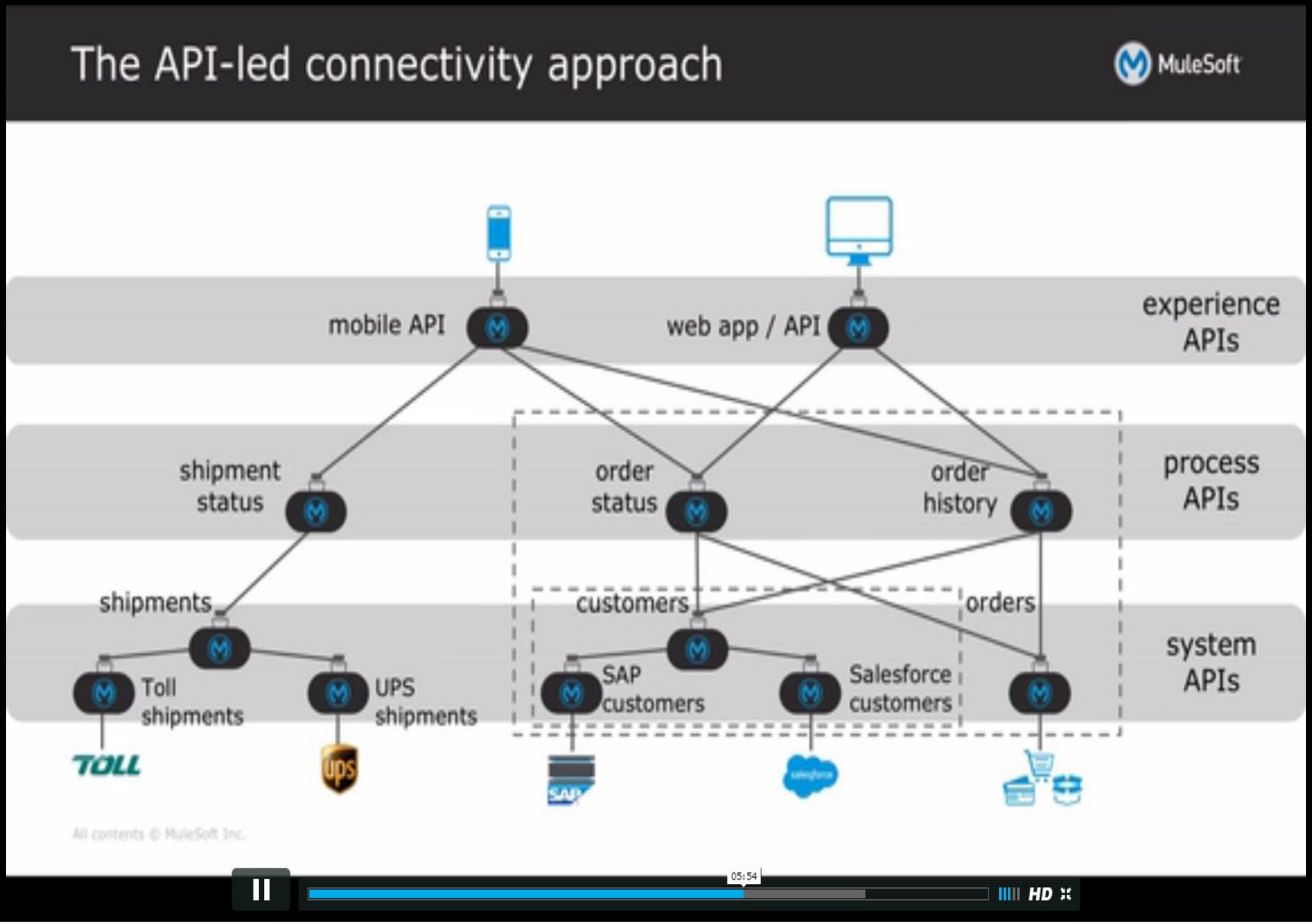
MuleSoft



The API-led connectivity approach

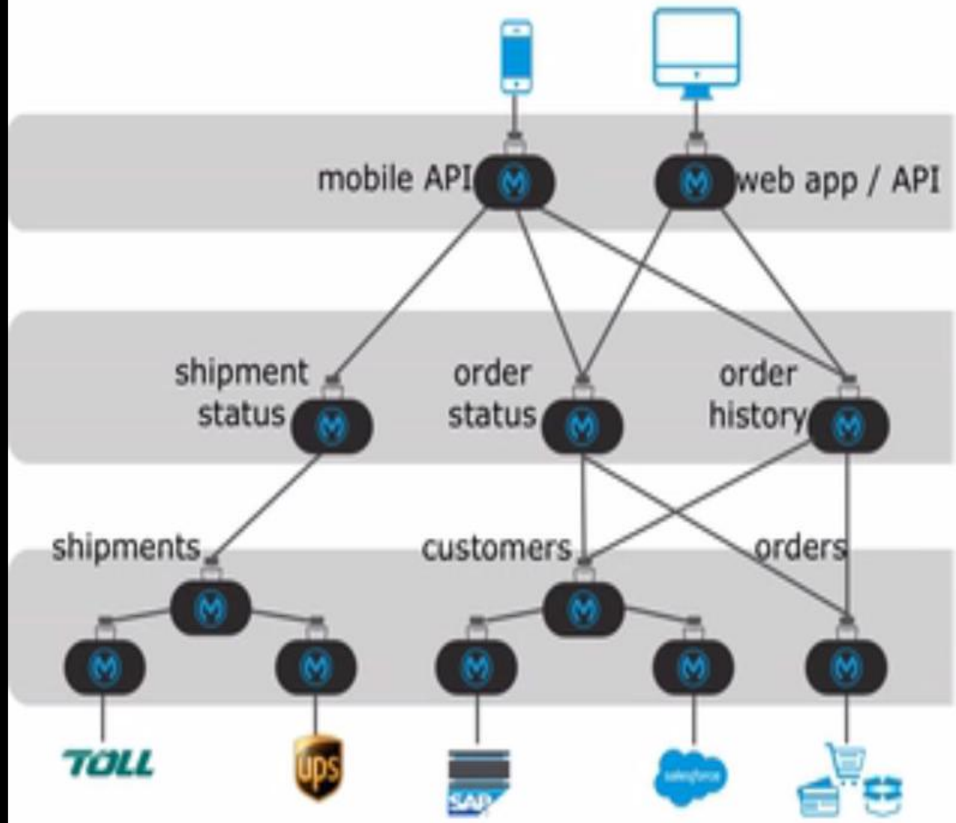


All work described was performed by Capgemini or a Capgemini affiliate



The API-led connectivity approach

MuleSoft



- On time
- Within budget
- Meets requirements
- Reusable components
- Loose coupling
- Governance built in

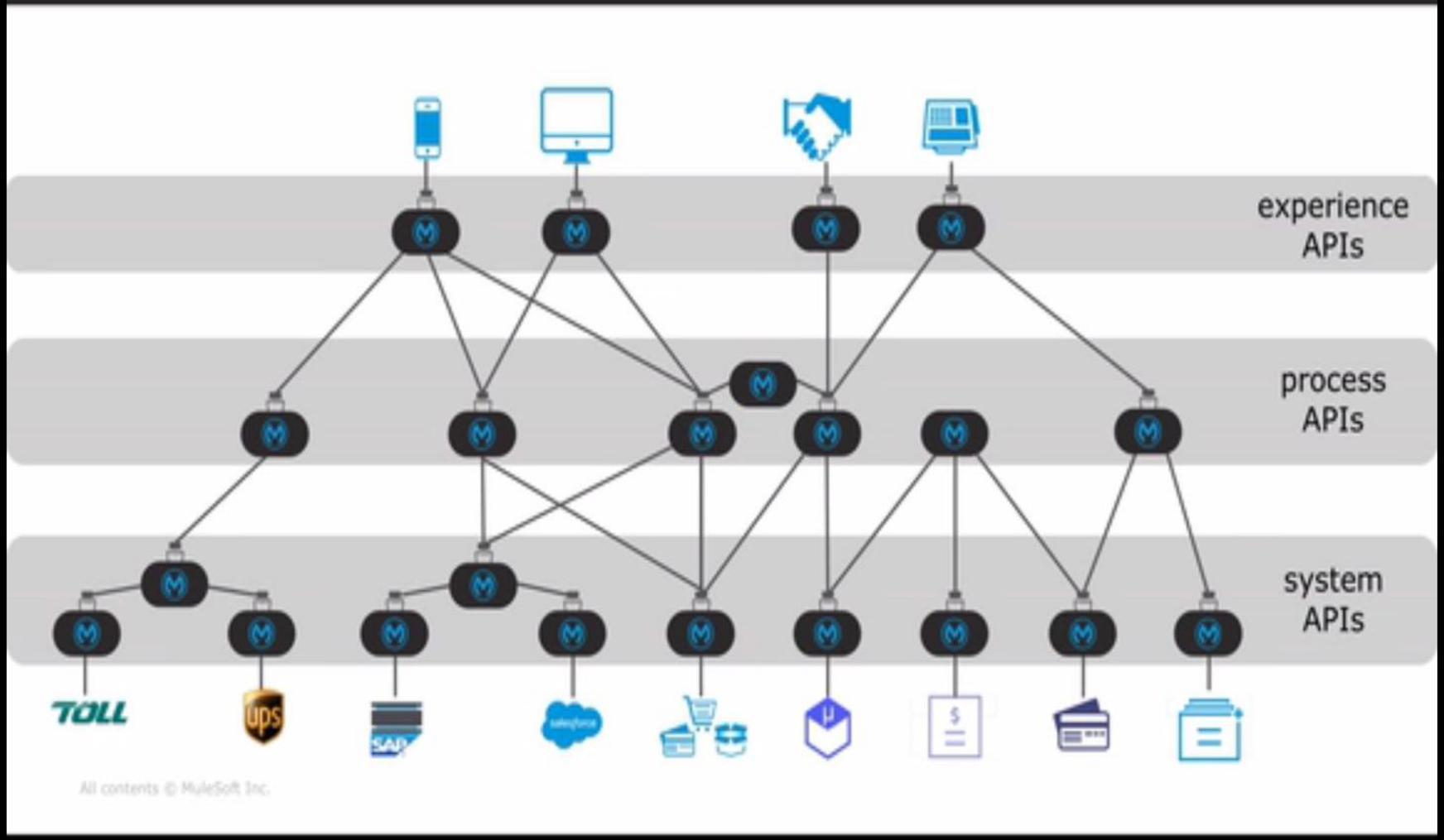
experience APIs

process APIs

system APIs

API-led connectivity: future-proof

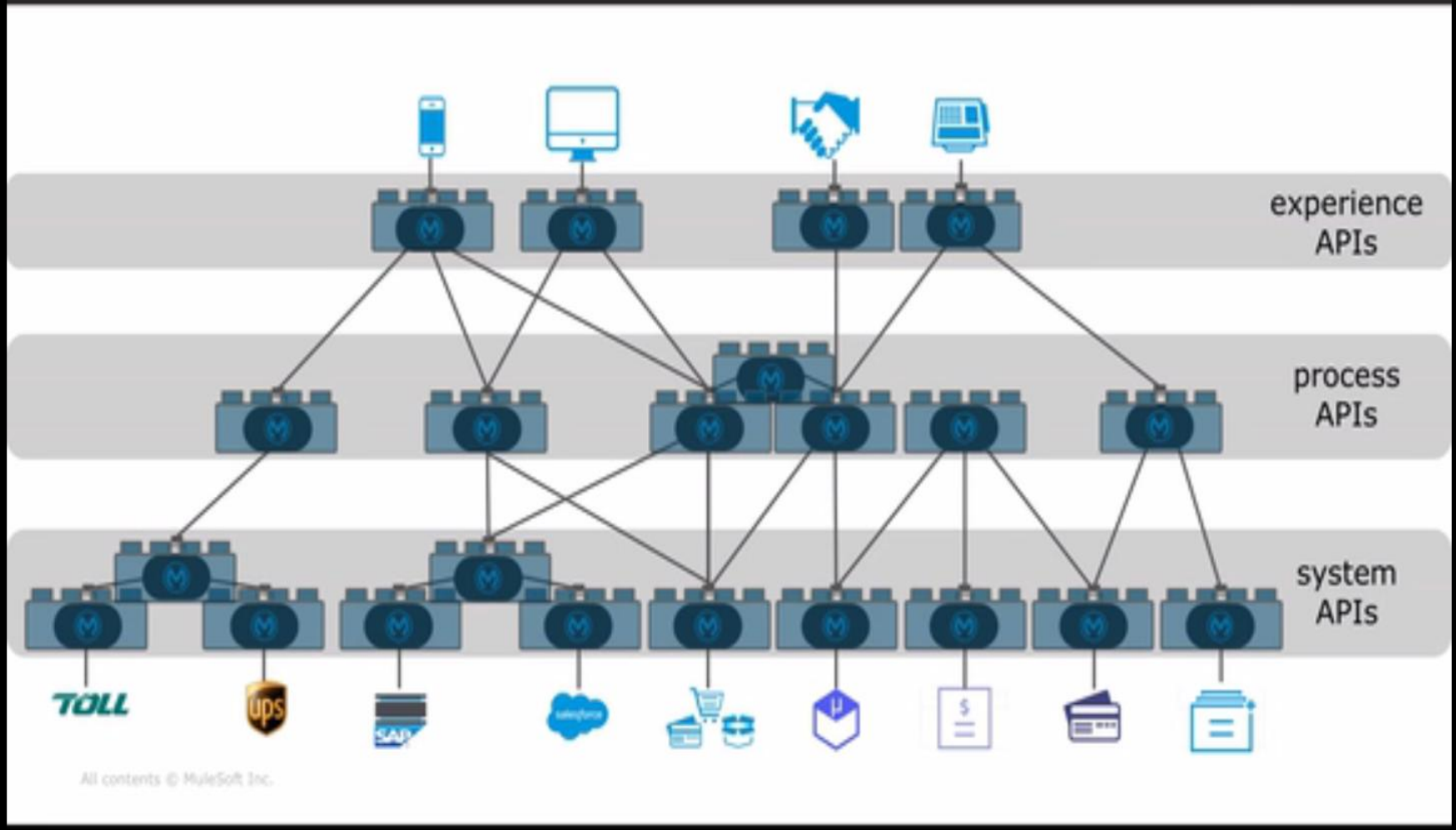
MuleSoft



All work described was performed by Capgemini or a Capgemini affiliate

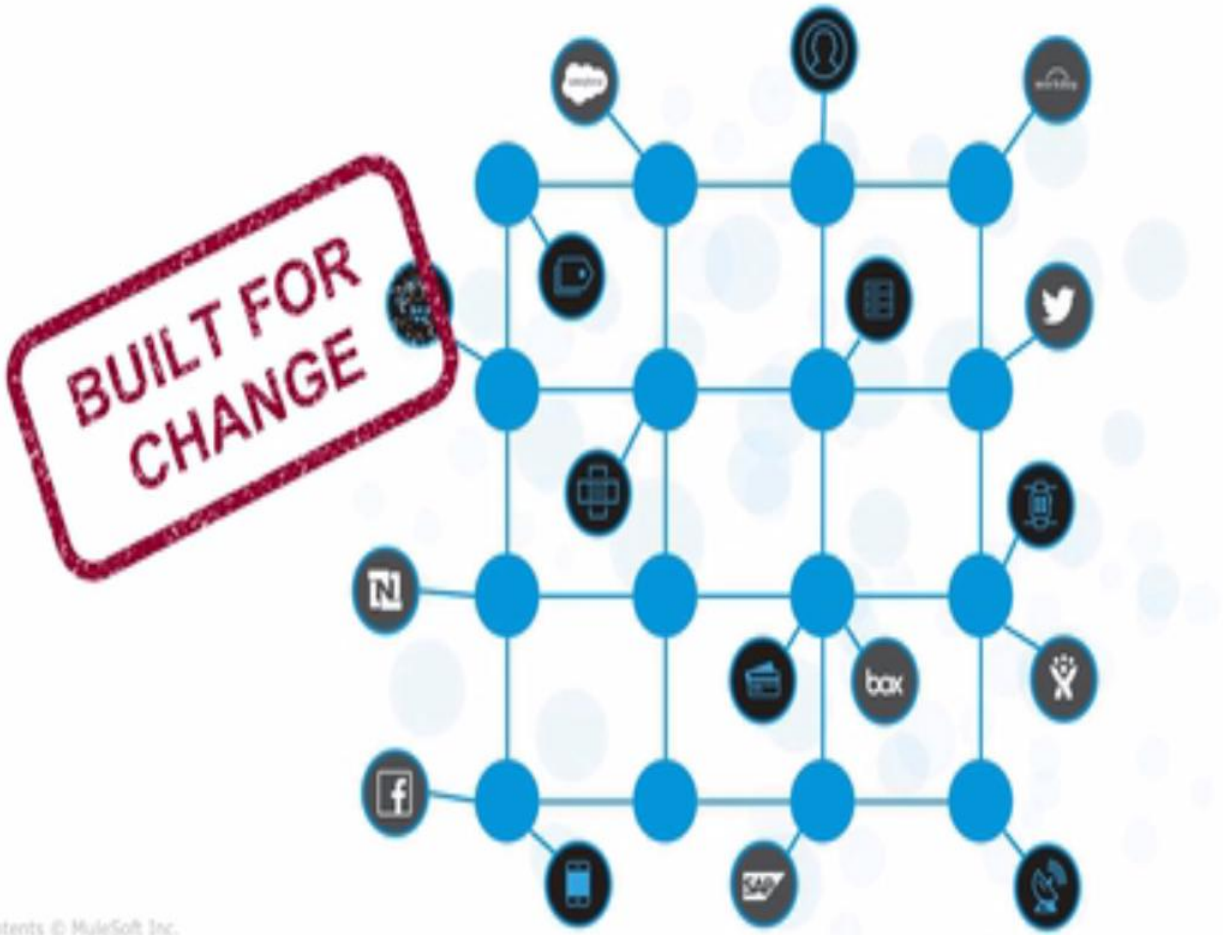
API-led connectivity: building blocks

MuleSoft



Your application network

MuleSoft



All work described was performed by Capgemini or a Capgemini affiliate

Recap

Words

important

highlighted

Thank You For Your Time

People matter, results count.