

TM Forum open API Fundamentals Student Notebook

© TM Forum

This document contains some key illustrations from the course to help you as you learn.

Please print this document and use it to take notes as you watch the videos. You should re-watch any material that you don't fully understand the first time through.

You should also study the TM Forum website where you will find documents that describe the TM Forum Open API Fundamentals in more detail.

The course and this document are not an exhaustive guide to the TM Forum Open APIs or the Framework.

Notice

This document is an annex to the TM Forum Open API Fundamentals Training Course.

No recipient of this document shall in any way interpret this document as representing a position or agreement of TM Forum or its members. It is a copyrighted document of TM Forum, provided for the sole use of the course student to make **one printed copy**. It is forbidden to make further paper or electronic copies for distribution.

Any use of this document by the recipient, other than as set forth specifically herein, is at its own risk, and under no circumstances will TM Forum be liable for direct or indirect damages or any costs or losses resulting from the use of this document by the recipient.

This document may involve a claim of patent rights by one or more TM Forum members, pursuant to the agreement on Intellectual Property Rights between TM Forum and its members, and by non-members of TM Forum.

Direct inquiries to the TM Forum office:
4 Century Drive
Suite 100
Parsippany, NJ 07054 USA
Tel No. +1 973 944 5100
Fax No. +1 973 944 5110
TM Forum Web Page www.tmforum.org

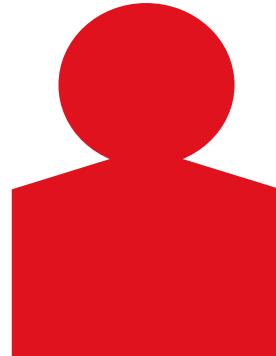
TM Forum Open API Fundamentals

R17.0_v1.0.1



Housekeeping

tmforum TRAINING



YOUR INSTRUCTOR

COURTESY

Please silence mobiles,
laptops down...

Q&A

Points of clarification
as they arise.
Other issues at
start/end of sessions
during the breaks/
end of day...

FEEDBACK

Please return
evaluation forms

Introductions

tm**forum** TRAINING

Name

Company

Professional background

What are your course expectations?

What is your experience?



Objectives

tmforum TRAINING

Understand the TM Forum Open APIs and the related technologies like REST

Learn how the TM Forum Open APIs are structured and know what are the existing and upcoming Forum Open APIs

Understand the TM Forum Open REST API Design Patterns and how all the Forum APIs are based on these patterns

Understand how the TM Forum Open APIs REST Resource Model is related to the Forum Information Framework

Understand Tooling and Forum Open API Conformance



Agenda

Introduction to REST APIs

TM Forum Open API Introduction

TM Forum Open API Structure

TM Forum Open API Design Patterns

TM Forum Open API and the Information Framework

Tooling, Sandbox and Conformance

Introduction to TM Forum Open API MicroServices

Course Wrap up



Acronyms

tmforum TRAINING



TM Forum Information
Framework



TM Forum Business
Process Framework



TM Forum Application
Framework



TM Forum Open APIs



TM Forum



Module 1:

Introduction to REST APIs



What are APIs?

Application Programming Interface

Provide interfaces to computer system for machine consumption

Allows applications
(software programs) to
talk to
one another



APIs are for Machine Consumption

tm**forum** TRAINING

Airline Reservation System



GUI



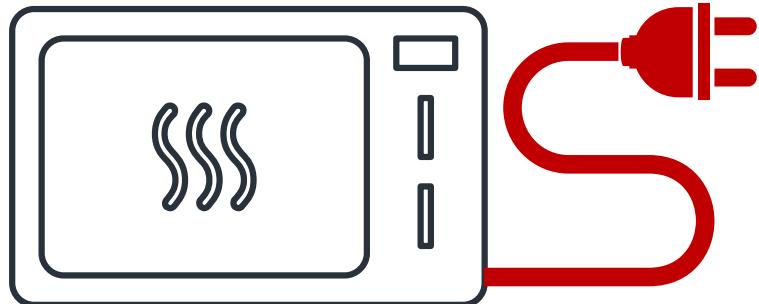
API

Machine – Application Software

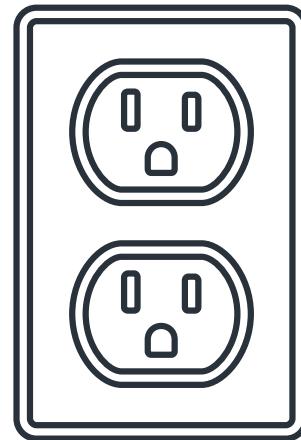


APIs are like Sockets!

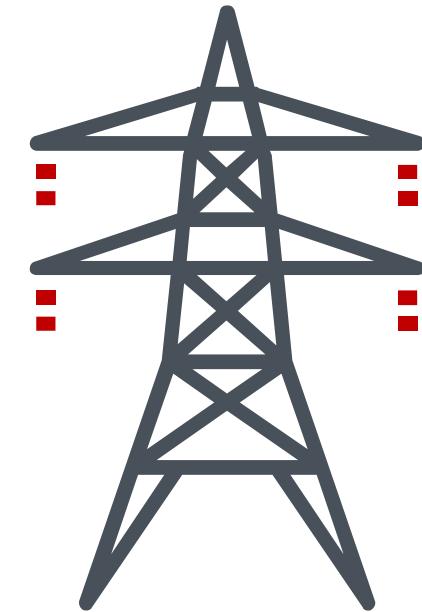
tm**forum** TRAINING



Consumer



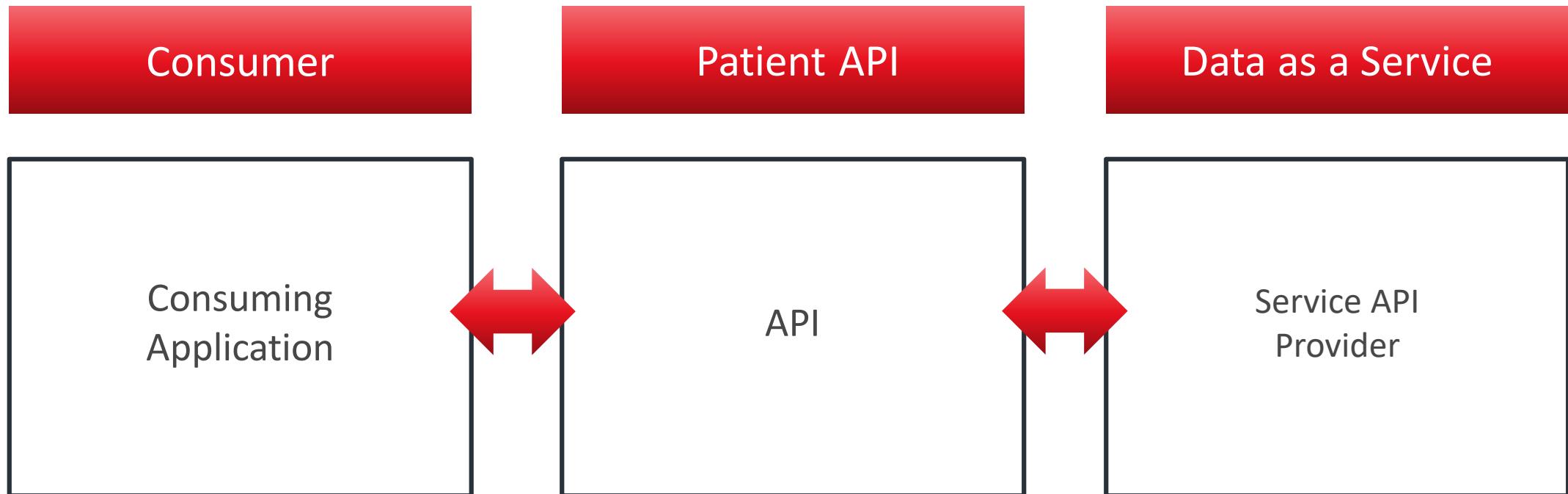
Wall Socket
Interface = API



Electricity
Service

APIs enable Applications

tmforum TRAINING



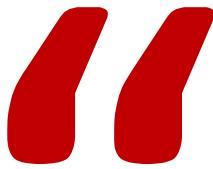
API is a Contract

tmforum TRAINING



Property of TM Forum.

For official TM Forum Training participant personal use/reference only.



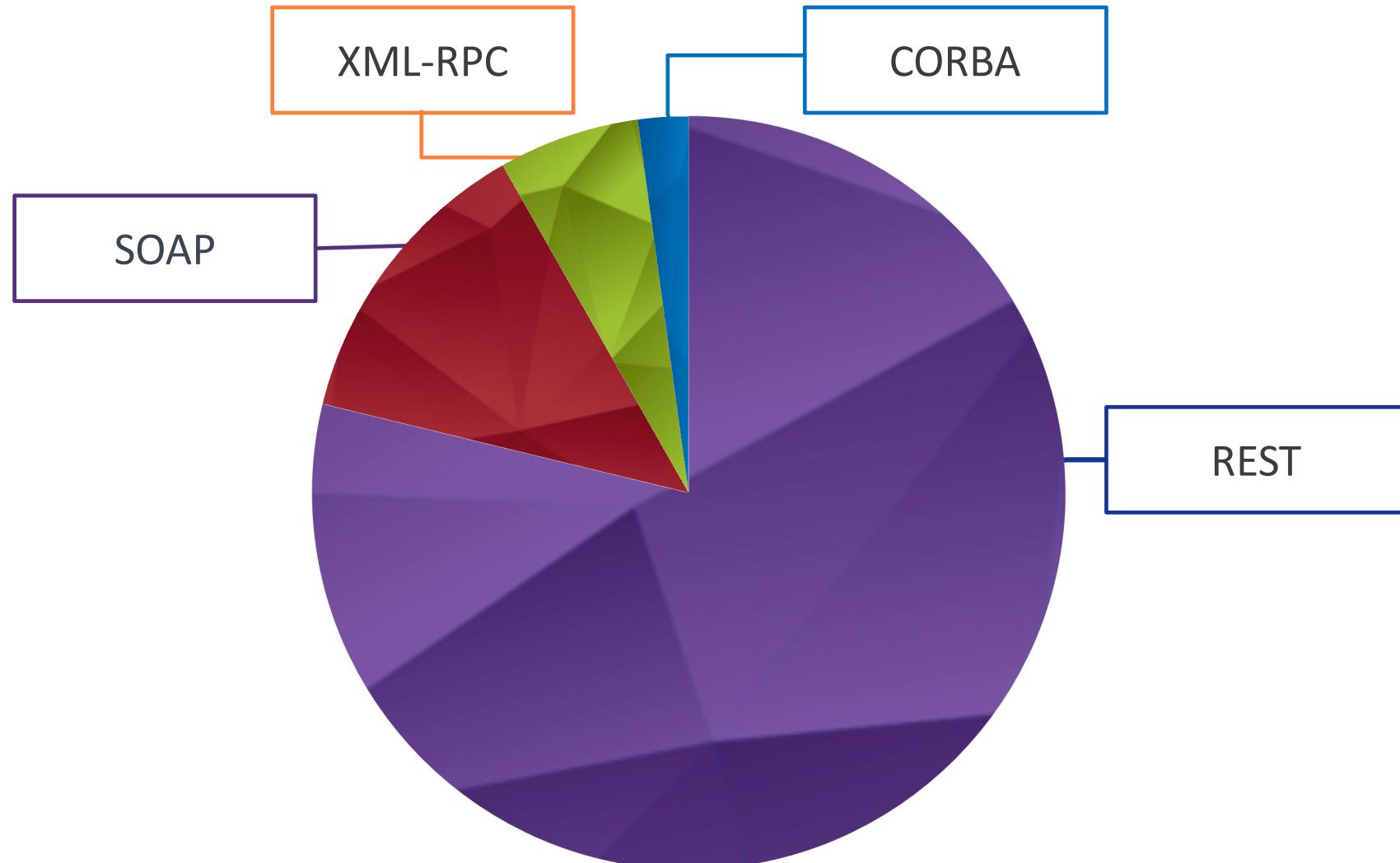
A server-side web API is a programmatic interface consisting of one or more publicly exposed endpoints to a defined request - response message system, typically expressed in JSON or XML, which is exposed via the web - most commonly by means of an HTTP-based web server.

Wikipedia



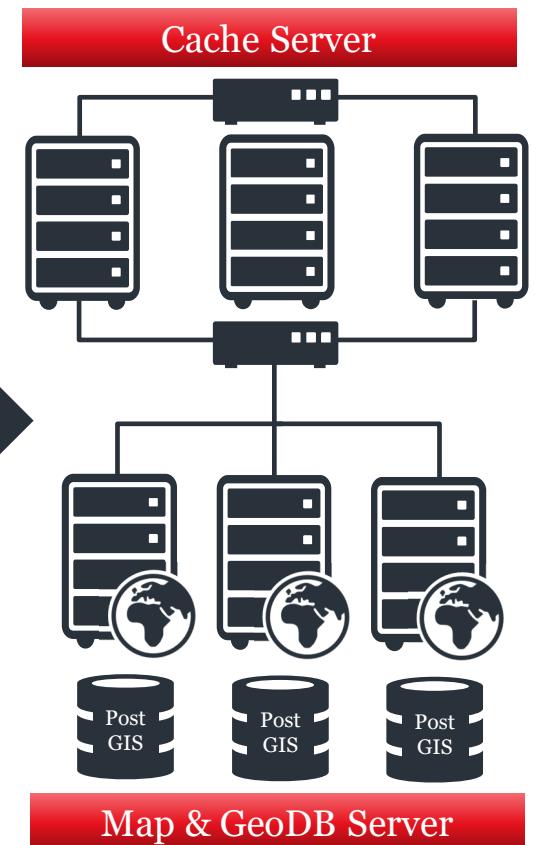
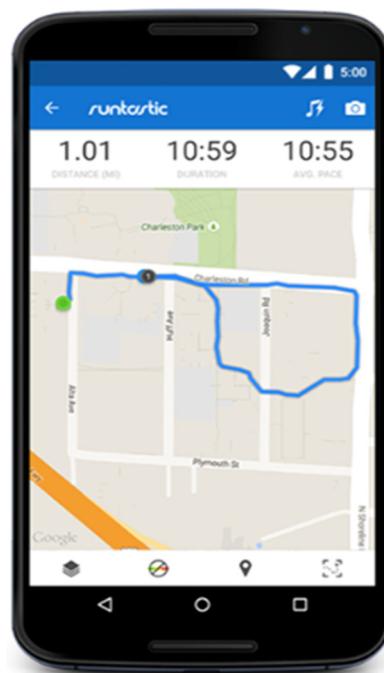
Web APIs Protocols and Styles

tmforum TRAINING



Google Map API Example

tm**forum** TRAINING



Two types of Web APIs

SOAP Web Services

use XML messages that follow the Simple Object Access Protocol (SOAP) standard

written in the Web Services Description Language (WSDL)

TM Forum TIP, MTOSI and OSS/J APIs

REST Web Services

better integrated with HTTP than SOAP-based services

defined by Swagger or RAML

payload in JSON (Java Scripting Object Notation)

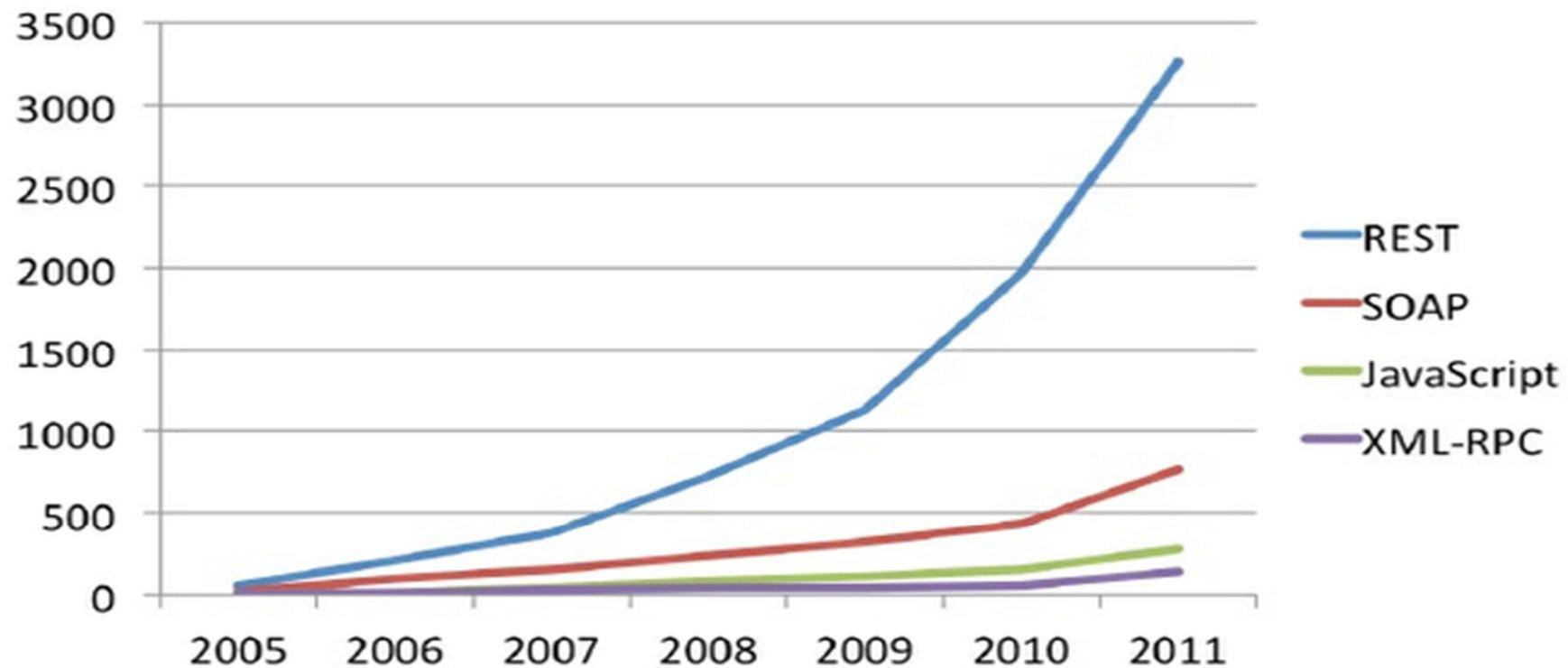
TM Forum Open APIs



REST versus SOAP?

tm**forum** TRAINING

API Protocols and styles



• Based on directory of 5,100 web API's listed at ProgrammableWeb, February 2012



REpresentational State Transfer

Architectural style of client-server applications not a technology

Centred around the transfer of representations of resources through requests and responses

Architecture is defined by REST constraints



Resources and Domain Entities

tmforum TRAINING

Resources represent the domain entities managed by the application

Resources are acted upon
by the API operations

Operations on Resources

A resource can be a trouble Ticket, a Logical Port , an Order, a Task etc...

Using the Uniform Contract Verbs POST, GET, PUT, DELETE, PATCH, etc....

Affect the lifecycle
and state of the
corresponding
entities



REST Constraints

Client-Server

Stateless

Cache

Interface/Uniform Contract

Layered System

Code-on-Demand



Everything is a Resource

Resources can have multiple representations

Suitable for CRUD (Create/Read/Update/Delete)

Link things together (Hypermedia)

Stateless Communication

Cacheable



REST APIs Today

REST does not require that a REST API be HTTP based

Most Web APIs today are REST based and use JSON as a representation format

REST APIs relies on a stateless, client-server, cacheable communications protocol -- and in virtually all cases today , the HTTP protocol is used

REST APIs use existing well-known W3C and Internet Engineering Task Force (IETF) standards (HTTP, XML, URI, MIME)



Uniform Contract Methods in HTTP

tm**forum** TRAINING

HTTP set of generic methods

GET, PUT, POST, PATCH, DELETE, HEAD
and OPTIONS

Pre-defined in the HTTP specification

Complete protocol interactions

A set of response codes

A set of error codes

Plus syntax for expressing various
parameters that can be encoded in
HTTP messages



REST API Operations

tm**forum** TRAINING

Operation on Entities	Uniform API Operation	Description
Query Entities	GET Resource	GET must be used to retrieve a representation of a resource
Create Entity	POST Resource	POST must be used to create a new resource
Partial Update of an Entity	PATCH Resource	PATCH must be used to partially update a resource
Complete Update of an Entity	PUT Resource	PUT must be used to completely update a resource identified by its resource URI
Remove an Entity	DELETE Resource	DELETE must be used to remove a resource
Execute an Action on an Entity	POST on TASK Resource	POST must be used to execute TASK resources
Other Request Methods	POST on TASK Resource	GET and POST must not be used to tunnel other request methods

[API Reference](#) [API Overview](#) [Release Notes](#) [eBay REST](#) [Selling APIs](#) [Integration Guide](#)

The **Inventory API** gives sellers the ability to create and manage offers for each of their products.

[Field Index](#) [Enumeration Index](#) [Type Index](#)

Call	Summary	Sample
INVENTORY_ITEM		
Bulk Update Price and Quantity POST /bulk_update_price_quantity	This call is used by the seller to update the total ship-to-home quantity of an inventory item and/or to update the price and/or quantity of that same inventory item in an actual live offer.	view
Create or Replace Inventory Item PUT /inventory_item/{sku}	This call creates a new inventory item record or updates an existing inventory item record.	view
Delete Inventory Item DELETE /inventory_item/{sku}	This call is used to delete an inventory item record associated with a specified SKU.	view
Get Inventory Item GET /inventory_item/{sku}	This call retrieves the inventory item record for a given SKU.	view
Get Inventory Items GET /inventory_item	This call retrieves all inventory item records defined for the seller's account.	view
INVENTORY_ITEM_GROUP		
Create or Replace Inventory Item Group PUT /inventory_item_group/{inventoryItemGroupKey}	This call creates a new inventory item group or updates an existing inventory item group.	view
Delete Inventory Item Group DELETE /inventory_item_group/{inventoryItemGroupKey}	This call deletes the inventory item group for a given inventoryItemGroupKey value.	view
Get Inventory Item Group GET /inventory_item_group/{inventoryItemGroupKey}	This call retrieves the inventory item group for a given inventoryItemGroupKey value.	view
LOCATION		
Create Inventory Location POST /location/{merchantLocationKey}	Creates an inventory location for the seller's account.	view
Delete Inventory Location DELETE /location/{merchantLocationKey}	Deletes the specified inventory location from the seller's account.	view
Disable Inventory Location POST /location/{merchantLocationKey}/disable	Disables the specified inventory location.	view
Enable Inventory Location POST /location/{merchantLocationKey}/enable	Enables the specified inventory location.	view
Get Inventory Location GET /location/{merchantLocationKey}	Retrieves the details of the specified inventory location.	view
Disable Inventory Location	Disables the specified inventory location.	view

eBay API Example with JSON

tm**forum** TRAINING

[API Reference](#) [API Overview](#) [Release Notes](#) [eBay REST](#) [Selling APIs](#) [Integration Guide](#)

The SKU value associated with the inventory item to retrieve is input into the end of the call URL. In this case, the SKU value for the inventory item is GP-Cam-01.

URL format. See also the [non-wrapped](#) version of this URL.

```
GET https://api.ebay.com/sell/inventory/v1/inventory_item/GP-Cam-01
```

Output

Based on the call response, the product identified as GP-Cam-01, has a total, ship-to-home quantity of 50.

The condition of the product is NEW and there are three images associated with the inventory item. The **product** container also consists of the product's title, description, and item specifics.

JSON format.

```
{
  "sku" : "GP-Cam-01",
  "availability":
  {
    "shipToLocationAvailability":
    {
      "quantity": 50
    }
  },
  "condition": "NEW",
  "product":
  {
    "title": "GoPro Hero4 Helmet Cam",
    "description": "New GoPro Hero4 Helmet Cam. Unopened box.",
    "aspects": {
      "Brand" :["GoPro"],
      "Type" : ["Helmet/Action"],
      "Storage Type" : ["Removable"],
      "Recording Definition" : ["High Definition"],
      "Media Format" : ["Flash Drive (SSD)"],
      "Optical Zoom" : ["10x"]
    },
    "imageUrls": [
      "http://i.ebayimg.com/images/i/182196556219-0-1/s-11000.jpg",
      "http://i.ebayimg.com/images/i/182196556219-0-1/s-11001.jpg",
      "http://i.ebayimg.com/images/i/182196556219-0-1/s-11002.jpg"
    ]
  }
}
```

PayPal API

tm**forum** TRAINING

PayPal Developer

Docs

APIs

Support

Search

Dashboard

REST APIs

Overview
Authentication & Headers
Errors

API Reference

Billing Agreements API
Billing Plans API
Identity API
Invoicing API
Payment Experience API
Payments API
Payouts API
Vault API
Webhooks API

Release Notes

SDKs
SDKs Quickstart
Service Integrations

NVP / SOAP APIs

Credentials
Endpoints
Currency Codes
Country Codes

Create agreement

POST

/v1/payments/billing-agreements

Creates a billing agreement. In the JSON request body, pass an `agreement` object with the name, description, start date, ID of the billing plan on which to base the agreement, payer, and shipping address.

Update agreement

PATCH

/v1/payments/billing-agreements/`agreement_id`

Updates details of a billing agreement, by ID. Details include the description, shipping address, start date, and so on.

Show agreement details

GET

/v1/payments/billing-agreements/`agreement_id`

Shows details for a billing agreement, by ID.

Bill agreement balance

POST

/v1/payments/billing-agreements/`agreement_id`/bill-balance

Bills an outstanding amount for an agreement, by ID. In the JSON request body, pass an `agreement_state_descriptor` object with the reason for the agreement state change and the agreement amount and currency.

Cancel agreement

Re-activate agreement

Quiz

What are APIs?

What is a Web API?

What is a REST API?

What style of API is mostly used on the Web used today?

What representation format is mostly used with REST APIs today?

Name 2 REST APIs widely used in the Web



Module 2:

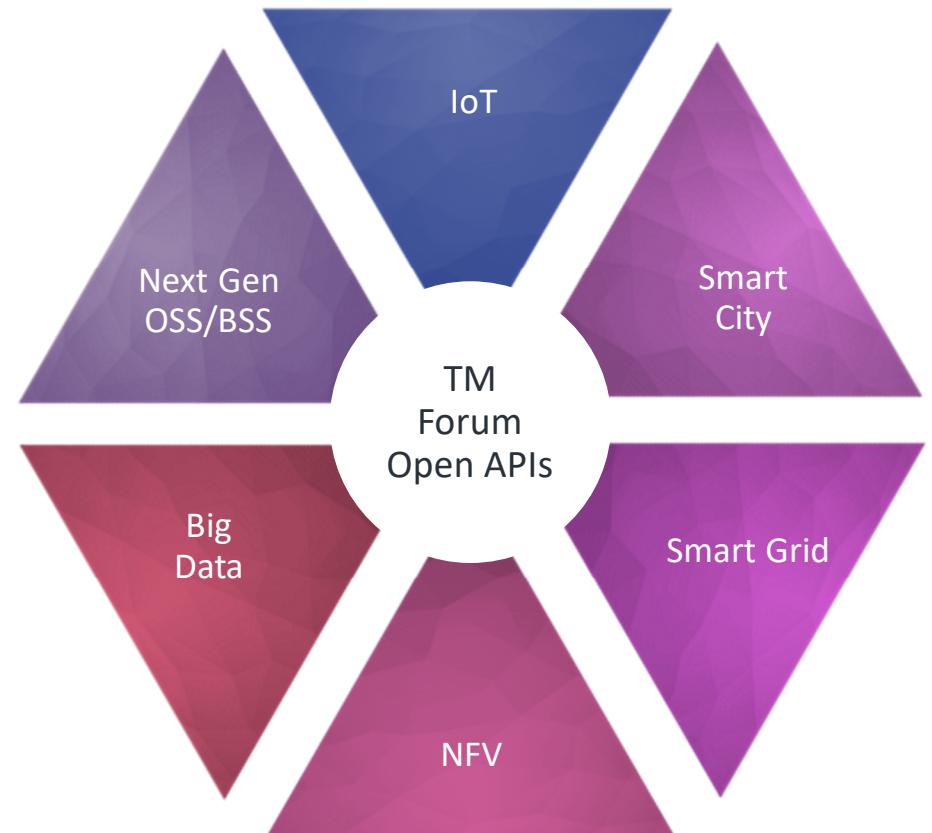
TM Forum Open API Introduction



What are the TM Forum Open APIs ?

tm**forum** TRAINING

- A suite of APIs making it easier to
 - create
 - build
 - and operate
- complex, innovative services
- REST based



- TM Forum Open APIs is a suite of APIs:
 - that enables services to be managed end to end throughout their lifecycle
 - in an environment where multiple partners are involved in the service delivery
- The APIs allow simple and coherent management of any given service
- The APIs are based on industry strength key design patterns to enable their rapid implementation.



Examples of Open APIs Currently Available

tmforum TRAINING

TM Forum Open APIs for Enabling Ecosystems

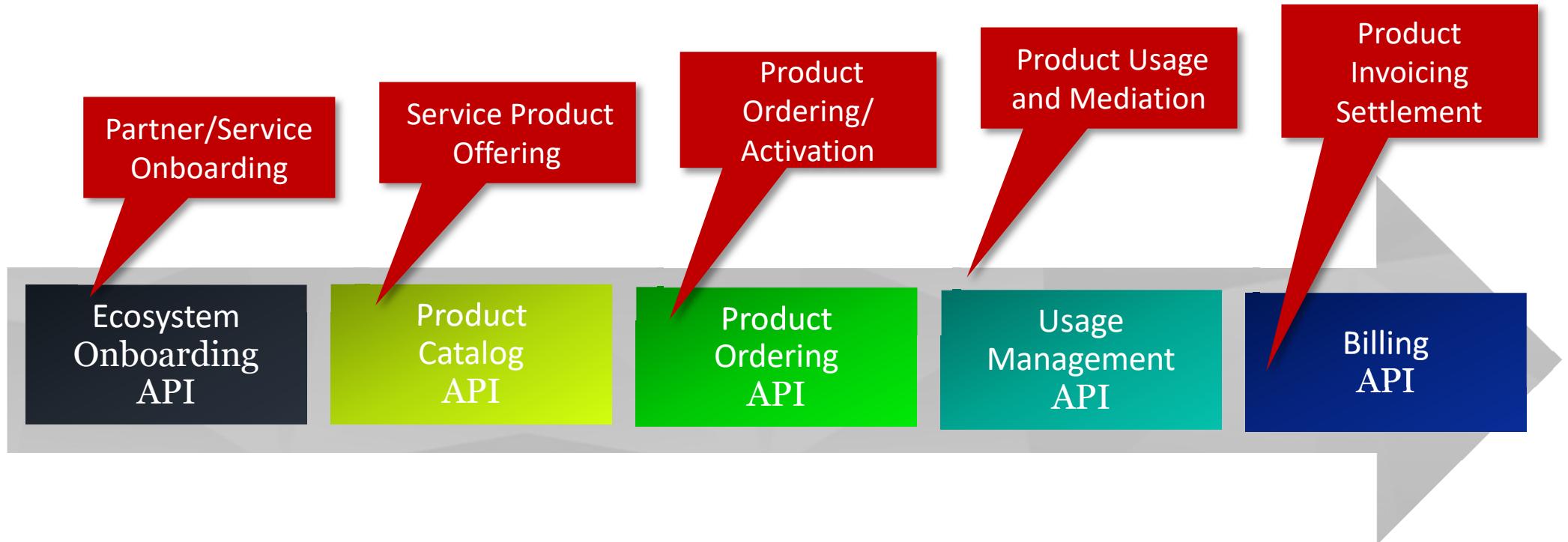


- From IoT device management to complex B2B value fabrics
- Managing partner arrangements across any business boundary
- Onboarding, SLA management, policy management and revenue sharing & settlement



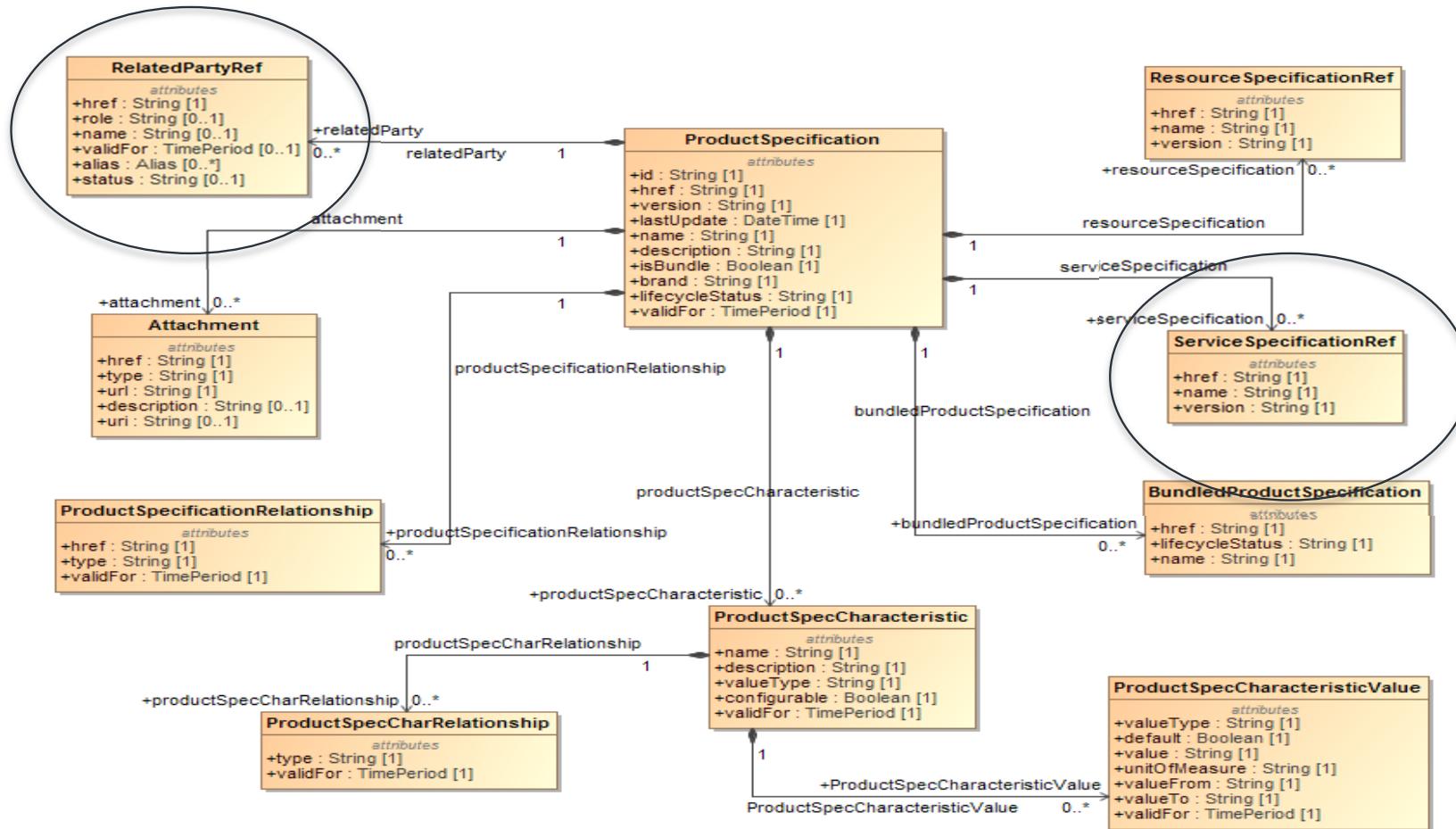
End to End Management

tmforum TRAINING



Simple Data Model with Hyperlinks...Derived from the SID tmforum TRAINING

Product Catalog API



TM Forum Open APIs are Service Agnostic

tmforum TRAINING

- Open APIs are generic framework APIs which can be used for any type of Product, Service and Resources
- Open APIs are generic framework APIs which can be used In the communications industry as well as other industries.
 - Network Connectivity Service
 - IOT Data Service
- Any Product, Service and Resources deployed in the TM Forum API Ecosystem can be instrumented with a set of generic APIs



Any Product can be Ordered via the Product Ordering API

Any Product can be Offered in a Product Catalog via Product Catalog API

Product underlying Services can be Activated and Provisioned via Service Activation and Provisioning API

Products can be Hyperlinked to Service Specifications in Service Catalogs via Product Catalog API

Products can be Hyperlinked to Service Level Agreement in SLA Management System supporting SLA APIs



Open APIs Supported by Full Specification and Test Kits

tmforum TRAINING

Released TM Forum Open APIs	API Specification	Postman Collection	Swagger Swagger_UI	Reference Implementation Code	Download Specification	CTK Profile
Trouble Ticket API Provides a standardized client interface to Trouble Ticket Management Systems for creating, tracking and managing trouble tickets among partners as a result of an issue or problem identified by a customer or another system. Examples of Trouble Ticket API clients include CRM applications, network management or fault management systems, or other trouble ticket management systems (e.g. B2B).						
Customer Management API Provides a standardized mechanism for customer and customer account management, such as creation, update, retrieval, deletion and notification of events.						
Product Catalog Management API Provides a standardized solution for rapidly adding partners' products to an existing Catalog. It brings the capability for Service Providers to directly feed partners systems with the technical description of the products they propose to them.						
Product Inventory Management API Provides standardized mechanism for product inventory management such as creation, partial or full update and retrieval of the representation of a product in the inventory. It also allows the notification of events related to product lifecycle.						

<http://www.tmforum.org/open-apis/>

<http://projects.tmforum.org/wiki/display/API/Open+API+Table>

Business agility and interoperability

By supporting an architecture based on modularity and reuse where capabilities are exposed through industry agreed Open APIs

Partnering for new services

Enable new products and services to be created by providing services to, and consuming services from, other business units, subsidiaries and external 3rd parties from telco and increasingly other verticals

Simplification of the IT estate

By enabling systems rationalisation

Reduced time to market

By enabling the configuration and reconfiguration of systems and capabilities through Open APIs to deliver new business solutions

Open APIs enabling Business Agility & Interoperability

tm**forum** TRAINING

40+ Open API specifications

28 Member Companies
Signed Open API Manifesto
committing to deploy TM Forum
Open APIs

14 Service Provider Markets
deployed & **27** by end of 2017

600+ Member Companies & **4000+**
Professionals downloading & using
the TM Forum Open APIs

7 Open Hacks **400+** Hack Participants
55 Innovative solutions

21,000+ Member Professionals



Signees of TM Forum's Open API Manifesto

Official API Manifesto

tm**forum** TRAINING

Open API Manifesto signed by 28 major companies since May 2016 committing to deploy TM Forum Open APIs



Signees of TM Forum's Open API Manifesto



Open API Roadmap

tmforum TRAINING

Current APIs

Open API Roadmap	Available	Planned Update
Activation & Configuration API	✓	
Address API	✓	Fx 17
Agreement API	✓	
Appointment API	✓	
Billing Management API	✓	
Change Management API	✓	Fx 17
Customer Management API	✓	
Entity Provisioning API	✓	
Loyalty Management API	✓	Fx 17
Onboarding API	✓	
Party Management API	✓	
Performance Management API	✓	Fx 17
Prepayment Balance Management API	✓	
Privacy API	✓	
Product Catalog Management API	✓	

Open API Roadmap	Available	Planned Update
Product Inventory Management API	✓	
Product Ordering API	✓	
Quote API	✓	
Resource Catalog API	✓	
Resource Ordering API	✓	
Service Catalog API	✓	
Service Inventory API	✓	
Service Ordering API	✓	
Service Problem Management API (SPM)	✓	
Service Quantification API	✓	
Service Quality Management API	✓	
Service Test Management API	✓	
SLA Management API	✓	
Trouble Ticket API	✓	
Usage Management API	✓	

TM Forum Open API Governance Activities

tm**forum** TRAINING

Management of the portfolio of APIs

Management of the individual API lifecycle

Management of policies and standards

Planning the development of new services and updating current services, through service submission and change requests

To ensure that service specification updates are consistent and interoperable through gated review and publication

The rules to which all APIs must conform should themselves be governed



Audience	TM Forum API Programme Team, including the contributor/author of the API specification
Input	All documents relevant to the API specification lifecycle milestone
Process	Consider the validity of the documents submitted against the principles and standards of the API programme, and relevant to the particular API development lifecycle milestone
Output	<ul style="list-style-type: none">• Approval: The documents are complete and correct. Approved for this milestone without need of further change.• Conditional Approval: The documents are approved to progress pending minor remedial actions that do not require a resubmission.• Resubmission: The documents require design rework and a fresh review.• Dispensation: The documents are allowed to progress with ‘technical debit’ – i.e.: acknowledging that they are not correct.• Escalation: Agreement cannot be reached within the governance board and requires a higher/wider authority.

API Governance Documents



Source	Description
API Governance Manifesto	12-point manifesto with very high-level guidance, and links to other material
	https://projects.tmforum.org/wiki/display/AP/Governance
GB990 API Data Model and Information Framework Mapping Guide Book R15.5.0	Describes SID->API Data Model mapping patterns (collapsing hierarchy, trimming relationships etc).
	https://projects.tmforum.org/wiki/display/AP/GB990+API+Data+Model+and+Information+Framework+Mapping+Guide+Book+R15.5.0
TR250 API REST Conformance Guidelines R15.5.0	"Self-Certification of TM Forum REST APIs". More relevant to API implementation
	https://projects.tmforum.org/wiki/display/AP/TR250+API+REST+Conformance+Guidelines+R15.5.0
API Governance	Wider Framework governance process, applied into API governance
	https://projects.tmforum.org/wiki/display/TP/API+Governance
TMF425 API Crowdsourcing Template R16.0.1	"the template for crowdsourcing APIs to TM Forum. It must be filled out prior to/in parallel with contributing a new API."
	https://projects.tmforum.org/wiki/display/PUB/TMF425+API+Crowdsourcing+Template+R16.0.1
TMF630 Conformance to REST API Design Guidelines	Part 1: Practical guidelines for RESTful API's naming, CRUD, filtering and notifications
	Part 2: Advanced guidelines for RESTful API's lifecycle management, polymorphism, common tasks
	https://projects.tmforum.org/wiki/display/API/API+Design+Guidelines

Open API Roadmap

tmforum TRAINING

Current APIs

Open API Roadmap	Available	Planned Update
Activation & Configuration API	✓	
Address API	✓	Fx 17
Agreement API	✓	
Appointment API	✓	
Billing Management API	✓	
Change Management API	✓	Fx 17
Customer Management API	✓	
Entity Provisioning API	✓	
Loyalty Management API	✓	Fx 17
Onboarding API	✓	
Party Management API	✓	
Performance Management API	✓	Fx 17
Prepayment Balance Management API	✓	
Privacy API	✓	
Product Catalog Management API	✓	

Open API Roadmap	Available	Planned Update
Product Inventory Management API	✓	
Product Ordering API	✓	
Quote API	✓	
Resource Catalog API	✓	
Resource Ordering API	✓	
Service Catalog API	✓	
Service Inventory API	✓	
Service Ordering API	✓	
Service Problem Management API (SPM)	✓	
Service Quantification API	✓	
Service Quality Management API	✓	
Service Test Management API	✓	
SLA Management API	✓	
Trouble Ticket API	✓	
Usage Management API	✓	

Quiz

Name 2 benefits of using the Forum Open APIs

What is Open API Crowdsourcing ?

What is the Open API Governance Board ?

Can anyone submit a proposal for a new Forum Open API ?

Are all the APIs already developed ?

Name 2 TM Forum Open APIs

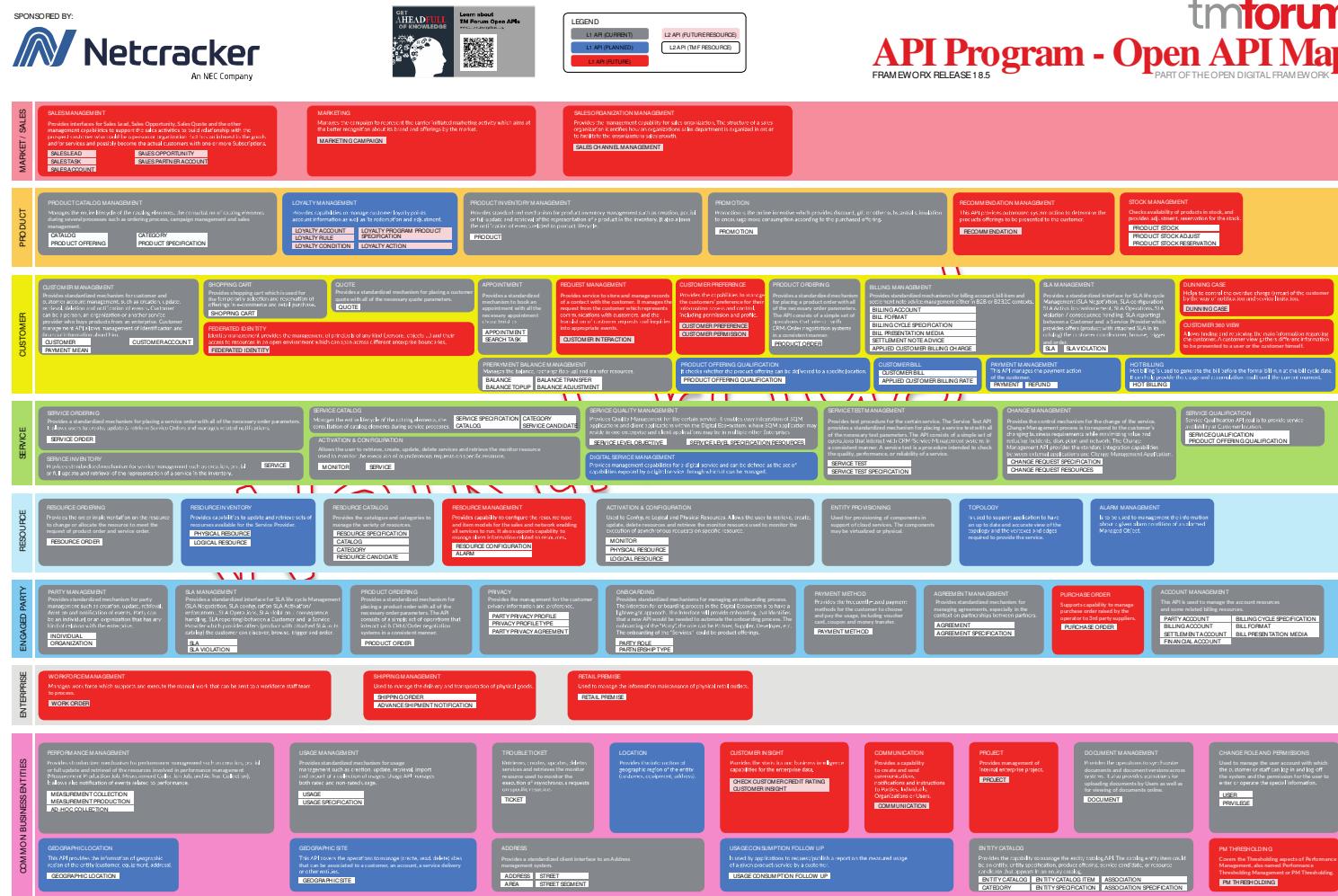


Module 3: TM Forum Open API Structure



API Structure

tmforum TRAINING



The Open API map is a work in progress. If you wish to contribute or comment on the map, join us on the Open API team calls. Poster design © NetCracker® and TM Forum 2019. File updated: April 17, 2020.

GB992 Open API Map R18.0.1 : <https://www.tmforum.org/resources/specification/gb992-open-api-map-r18-0-0/>
GB992 AI poster : <https://www.tmforum.org/resources/standard/open-api-poster-2019/>

Property of TM Forum.
For official TM Forum Training participant personal use/reference only

Open API Structure – Level 0

tmforum TRAINING

Market/Sales

Product

Customer

Service

Resource

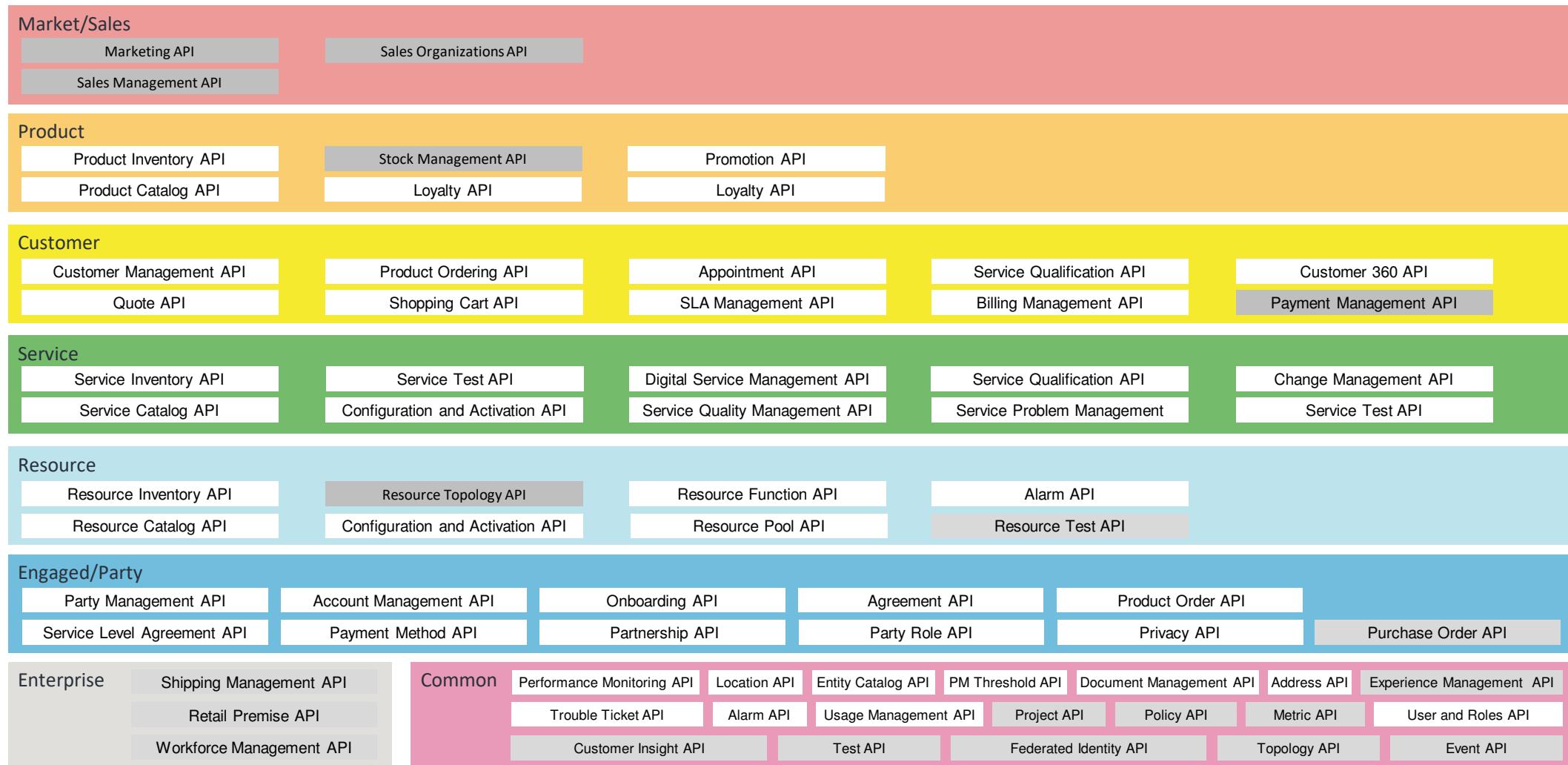
Engaged/Party

Enterprise

Common

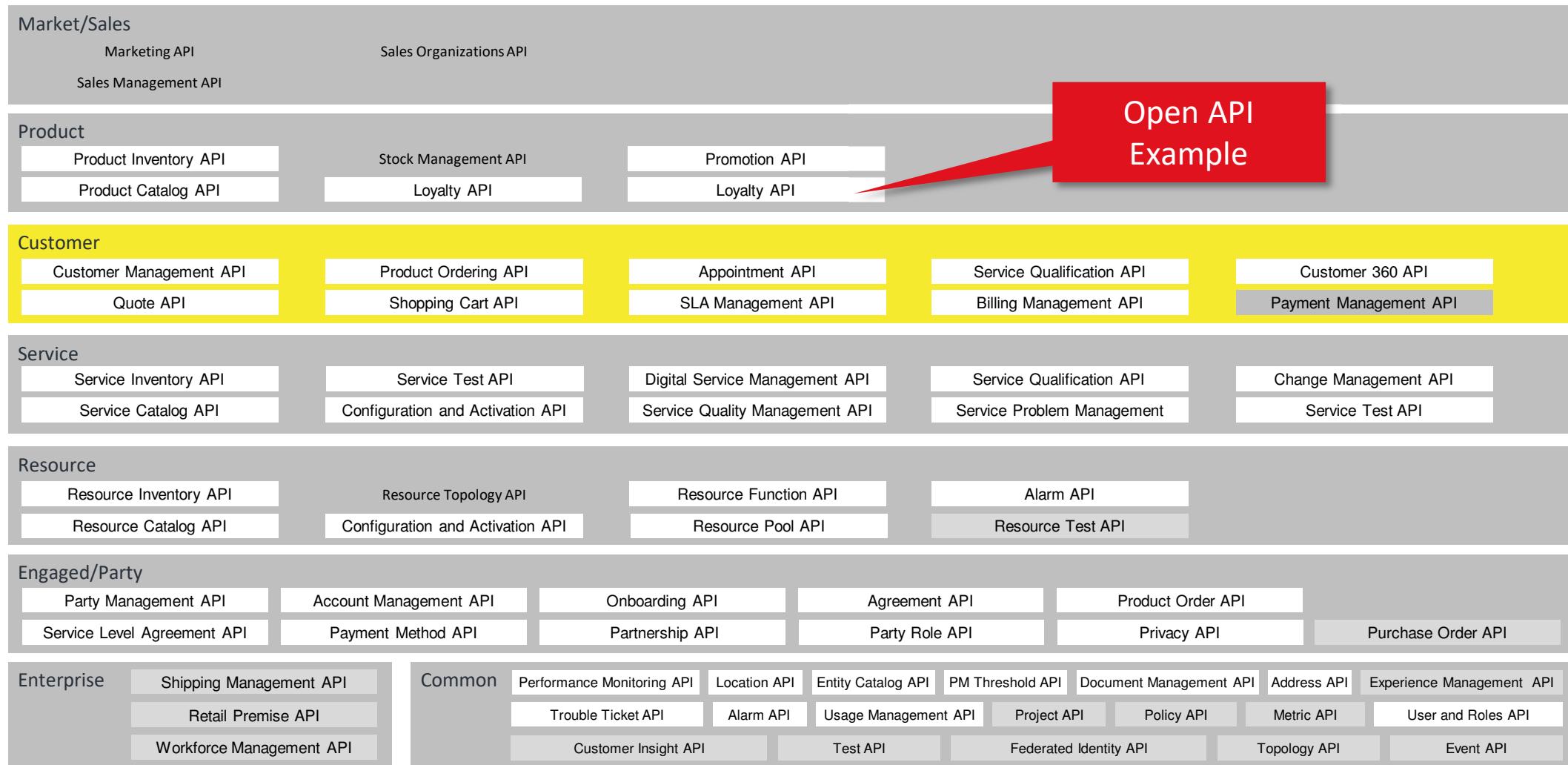
Open API Structure – Level 1

tmforum TRAINING



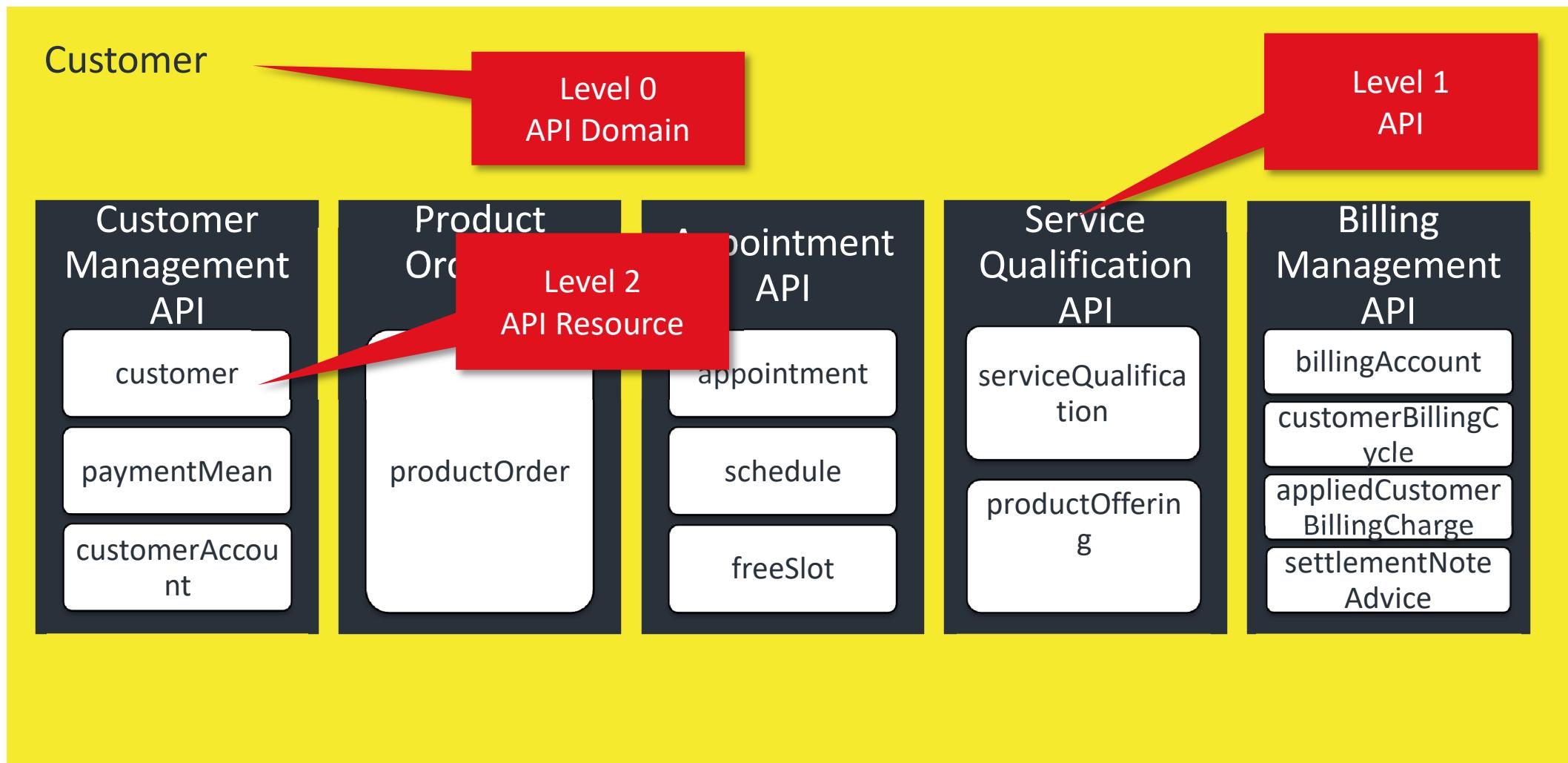
Open API Structure – Level 1

tmforum TRAINING



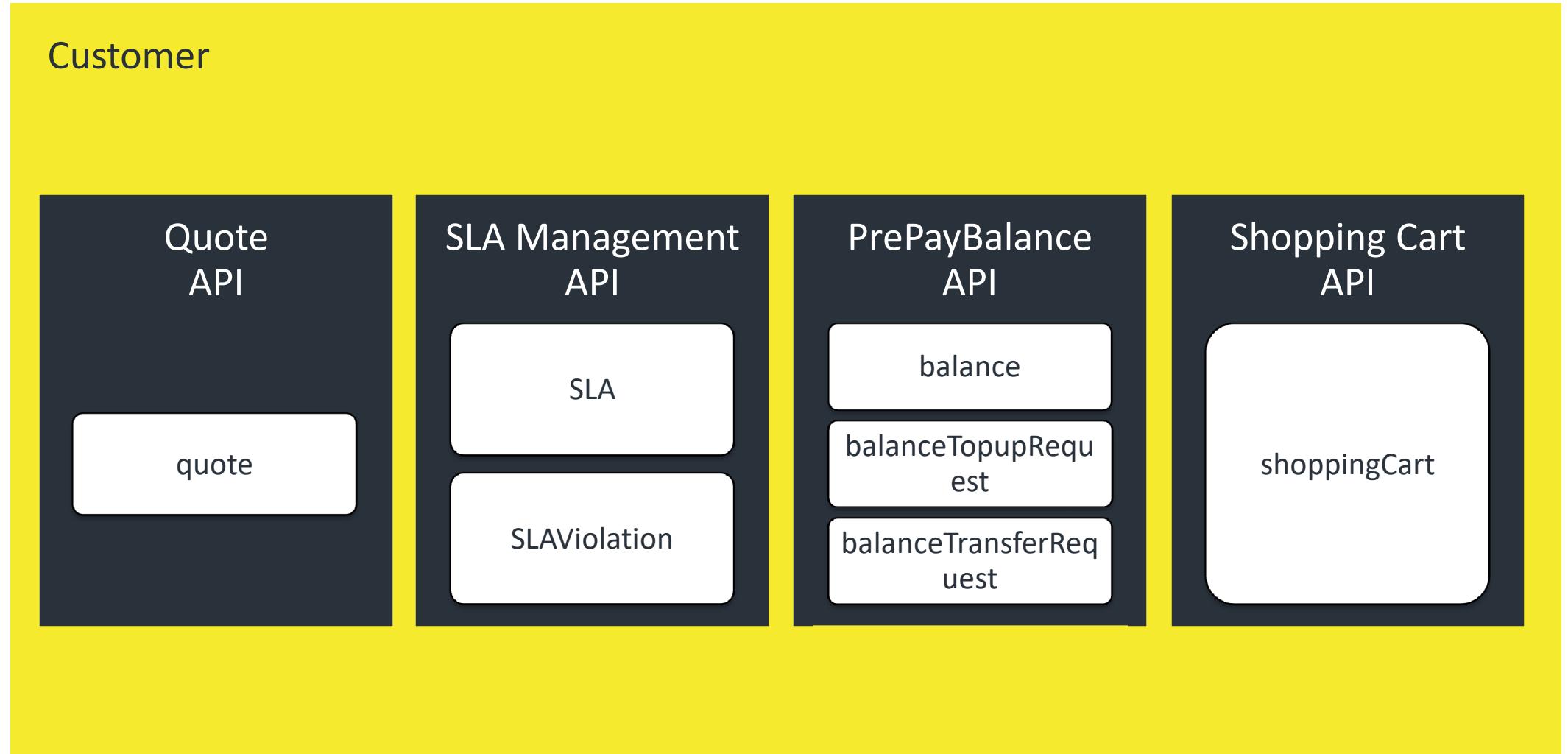
Open API Structure - Level 2 Customer Domain

tmforum TRAINING



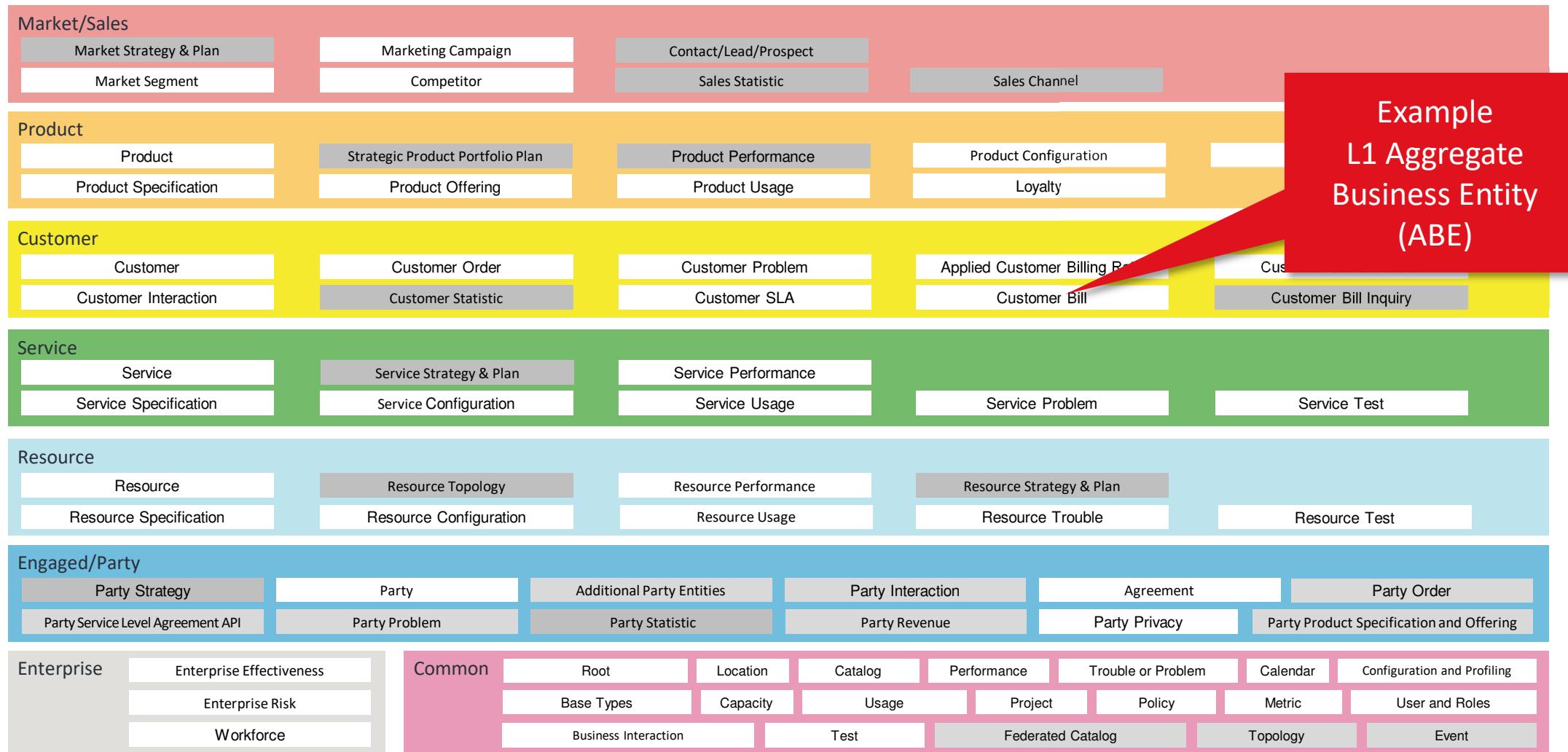
Open API Structure - Level 2 Customer Domain

tm**forum** TRAINING



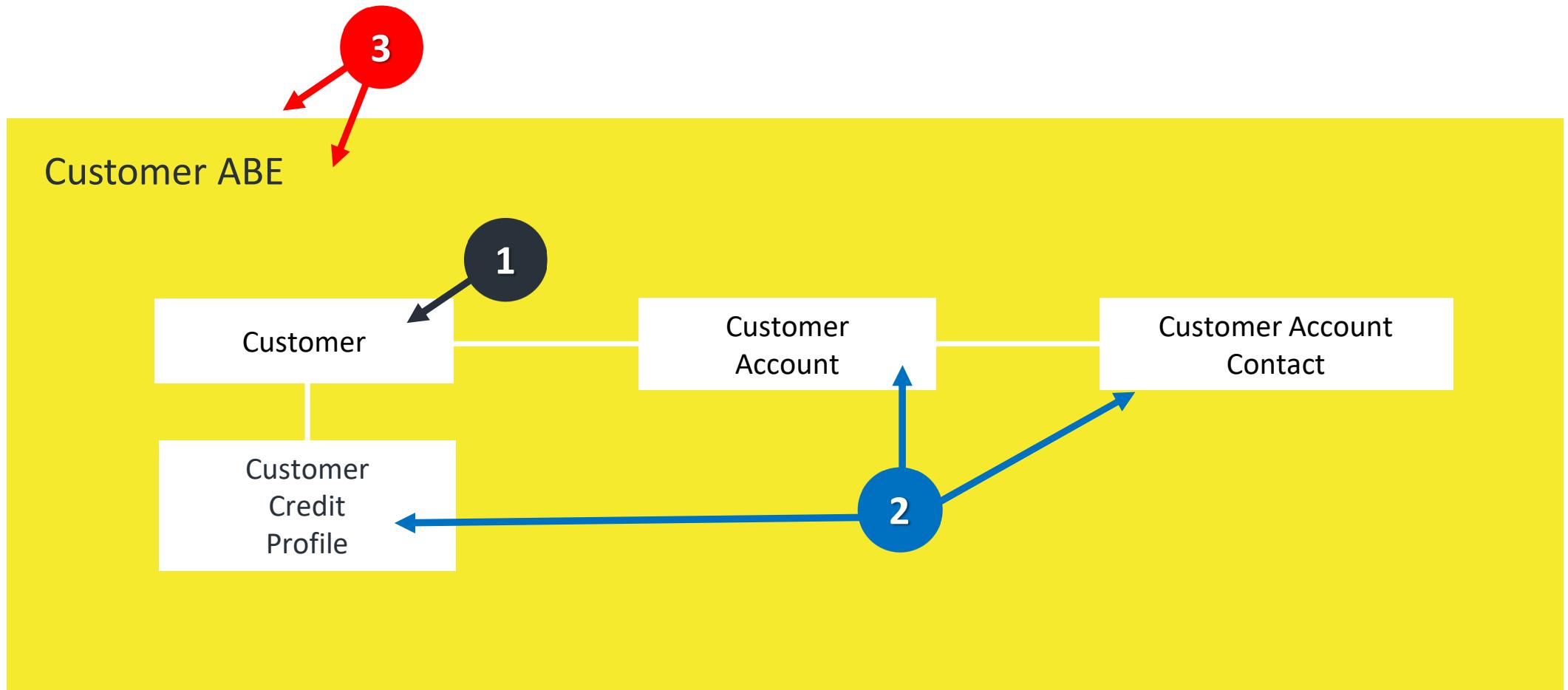
Information Framework Structure

tmforum TRAINING



Structure of an ABE

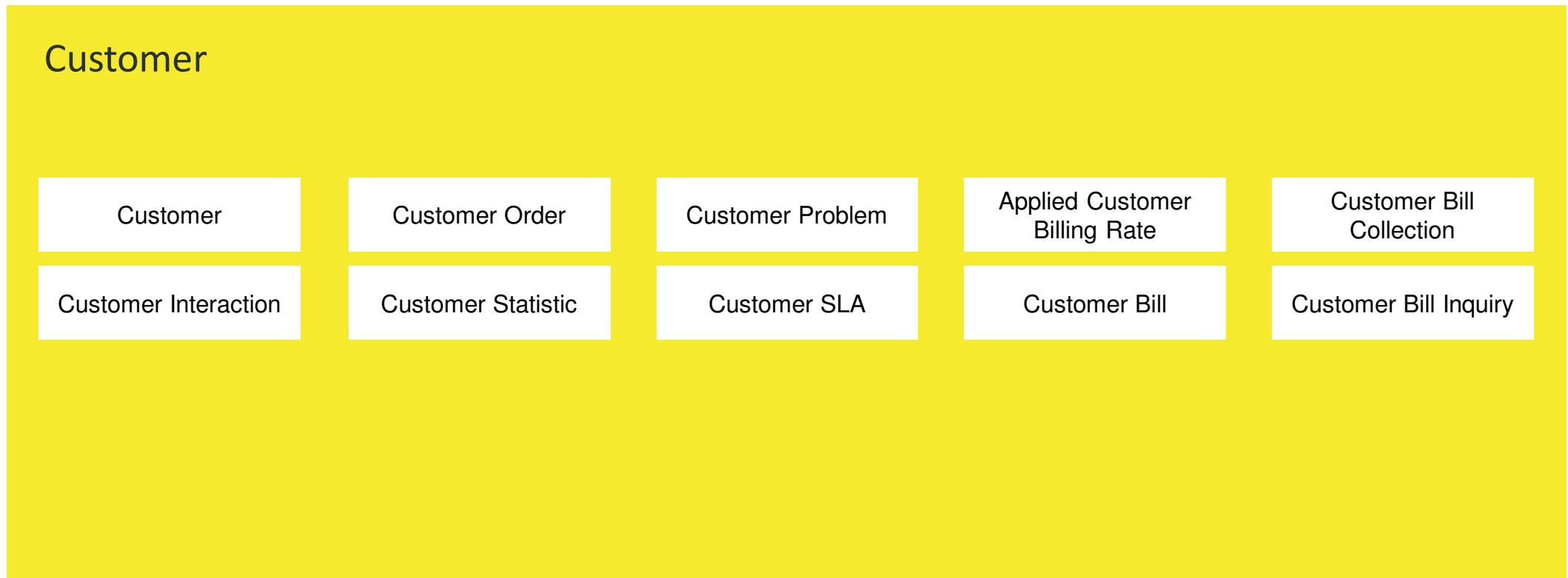
tmforum TRAINING



Structure of a Domain

tm**forum** TRAINING

All ABEs that completely define a key management area are grouped together



Mapping to SID ABE

tmforum TRAINING

#	API NAME (API)	Resource (API)	Information Framework Domain::L1 ABE-	API Lo Domain
1	Product Catalog Management API	Catalog	Common Business Entities Domain::Catalog ABE	Product
		Category	N/A	
		Product Offering	Product Domain::Product Offering ABE	
		Product Specification	Product Domain::Product Specification ABE	
2	Product Inventory Management API	Product	Product Domain::Product ABE	Product
3	Billing Management API	Billing Account	Customer Domain::Customer ABE Engaged Party Domain::Additional Party Entities ABE	Customer

Relationship with Frameworks

tm**forum** TRAINING

The TM Forum APIs are related to all other Forum Frameworks, in that they:

Are typically entity-centric, manipulating a primary resource that can be found in TM Forum Information Framework (SID), an Information view of the APIs

Play a part in (potentially a number of) business processes from TM Forum Business Process Framework (eTOM), a Business Process view of the APIs

Can be exposed by an normalized endpoint application or application-domain described by TM Forum Application Framework (TAM), an Application view of the APIs



API verification via Business Process

tmforum TRAINING

API verification
via Business Process
GB921D

GB992 Addendum
Mapping between
API and Business
Process
Framework

The Level 3 of TM
Forum Business
Process Framework
is chosen as the
benchmark of API
verification.



APIs and Business Process Framework

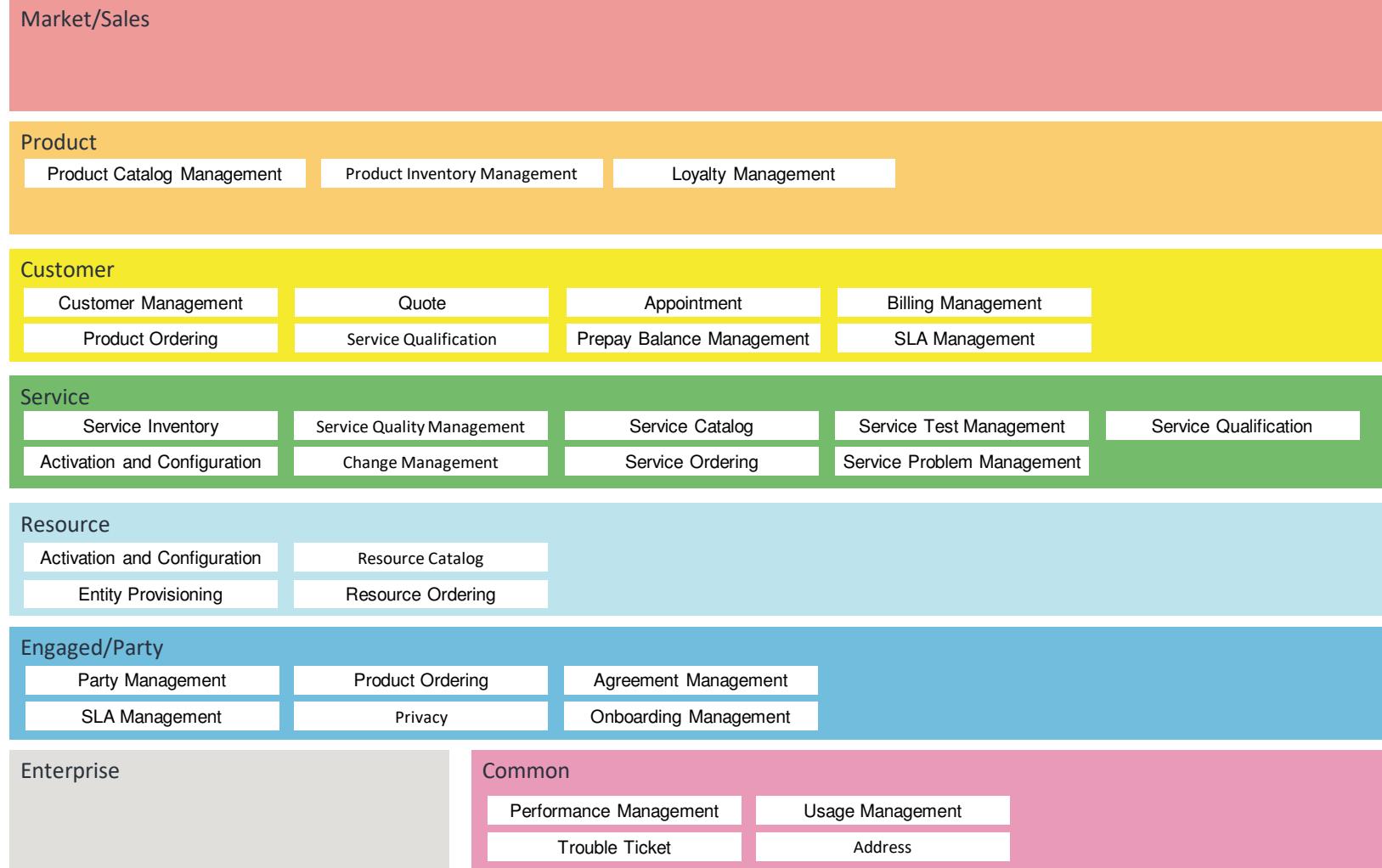
tmforum TRAINING

Framework Process	Level 3 Category	Process Identifier	Brief Description	TMF API Alignment	Process as API consumer or producer	TMF API Resource Name
Product Specification Development & Retirement	(3) eTOM Process Type	1.2.7.1	Develop and deliver new product specifications as well as enhancements and new features, ready for use by other processes, including Product Offering	Product Catalog Management Product Inventory Mgmt	Producer	Product Offering, Product
Product Offering Development & Retirement	(3) eTOM Process Type	1.2.7.2	Develop and deliver new product offerings, their pricing, as well as catalogs that contain both.	Product Catalog Management Product Inventory Mgmt	Producer	Product Offering, Product
Develop Product Capacity	(3) eTOM Process Type	1.2.8.1	Perform all applicable generalized capacity management processes associated with product capacity.	Product Catalog Management Product Inventory Mgmt	Producer	Product Offering, Product
Develop Product Capacity Demand	(3) eTOM Process Type	1.2.8.2	Perform all applicable generalized capacity management processes associated with product capacity demand.	Product Catalog Management Product Inventory Mgmt	Producer	Product Offering, Product
Support Customer Interface Management	(3) eTOM Process Type	1.3.1.1	Ensure that all information, materials, systems and resources are available so that the Customer Interface Management processes can operate	Customer Management		

Walkthrough

Overview of existing APIs – R17.0

tmforum TRAINING



Legend:
L1 API
(Current)

Quiz

tmforum TRAINING

What document defines the API Structure?

How many levels does the API MAP support?

Where can we find the REST resources exposed by an API?

What document describes the relationship between the Open APIs and the Forum Business Processes?



Module 4: TM Forum Open API Design Patterns



Why REST API Design Patterns ?

tm**forum** TRAINING



RESTful web services do not have any strict specifications for designing and building APIs

REST is a software architectural style, not a specification, for building scalable web services

Many interpretations exist for how the RESTful Web API should work

Need standards, best practices, conventions, and tips and tricks that you can apply to your RESTful web service applications

What are the TMF REST Design Patterns ?

tm**forum** TRAINING

- All TMF APIs are based on a common set of Patterns
- Patterns are documented in the TMF REST Guidelines (also known as TMF 630 and TMF 631)
- Patterns provide answers to recurrent questions in designing REST APIs

Patterns for using Uniform Contract operations for the implementation of Entity CRUD operations and Task operations.

Patterns on filtering and attribute selection.

Patterns for supporting Notification Management in REST based systems.

Versioning Patterns



API Resource Archetypes

Resource Collection, Managed Resource, Tasks

Uniform Contract Elements

Resource identifier syntax , Methods, Media types
All API operations are based on the REST Uniform Contract operations (GET,POST, DELETE, PUT, PATCH, HEAD, etc.)

Query Entities Design Patterns

Patterns used to query resources
Uniform Contract GET, Attribute selection, Attribute Filtering

Create Entities Design Patterns

Uniform Contract POST, Mandatory Attributes, Structure of returned Response, Location Header, Exception Code

Update Entities Design Patterns

Uniform Contract PATCH, Mandatory Attributes, Structure of returned Response, Location Header, Exception Code

Delete Entities Design Patterns

Uniform Contract DELETE, Mandatory Attributes, Structure of returned Response, Location Header, Exception Code

Tasks Design Patterns

A TASK resource name is a verb representing the task to be executed. TASK creation may result in the creation of a number of related resources or other tasks

Notification Design Patterns

Publish subscribe pattern
Hub and event Filtering

Asynchronous Operations Design Patterns

In some use cases the modification of an attribute may result in the execution a long running transaction before the attribute actually is changed.
TransactionMonitor resource is returned so that the execution of the operation can be monitored

Patterns for CRUD and Tasks

tmforum TRAINING

Operation on Entities	Uniform API Operation	Description
Query Entities	GET Resource	GET must be used to retrieve a representation of a resource
Create Entity	POST Resource	POST must be used to create a new resource
Partial Update of an Entity	PATCH Resource	PATCH must be used to partially update a resource
Complete Update of an Entity	PUT Resource	PUT must be used to completely update a resource identified by its resource URI
Remove an Entity	DELETE Resource	DELETE must be used to remove a resource
Execute an Action on an Entity	POST on TASK Resource	POST must be used to execute TASK resources
Other Request Methods	POST on TASK Resource	GET and POST must not be used to tunnel other request methods

TM Forum REST API Design Pattern Examples



Property of TM Forum.

For official TM Forum Training participant personal use/reference only.

Report Management Example

tm**forum** TRAINING

```
{  
    "id": << 42 >>,  
    "datetime": "2012-12-20 15:00:00",  
    "health_state": "operational",  
    "metrics": [  
        {  
            "code": "123",  
            "category_id": "45",  
            "date_time": "2012-12-20 14:30:00",  
            "reference": "4321",  
            "source_id": "8f27ce50-4b00-11e2-bcf0-0800200c9a66",  
            "value": 50,  
            "metric_id": "b0dbbc50-4b00-11e2-bcf0-0800200c9a66"  
        },  
        {  
            "code": "321",  
            "category_id": "48",  
            "date_time": "2012-12-20 14:30:00",  
            "reference": "4321",  
            "source_id": "8f27ce50-4b00-11e2-bcf0-0800200c9a66",  
            "value": 50,  
            "metric_id": "c50f3e90-4b00-11e2-bcf0-0800200c9a66"  
        }  
    ],  
    "failures": [  
        {  
            "failure_id": "0bec4420-4b01-11e2-bcf0-0800200c9a66",  
            "detail": "Network authentication failure",  
            "source_id": "24b2bc00-4b01-11e2-bcf0-0800200c9a66"  
        },  
        {  
            "failure_id": "3b687c50-4b01-11e2-bcf0-0800200c9a66",  
            "detail": "Network authentication failure",  
            "source_id": "462f1f40-4b01-11e2-bcf0-0800200c9a66"  
        }  
    ]  
}
```

How to select a subset of the attributes of an entity to be present in a returned representation?

- An attribute selector directive called “**fields**” is used to specify the attributes to be returned as part of a partial representation of a resource.

```
GET {apiRoot} /{resourceName}/{resourceID}/?fields={attributeName*}
```

- Used to retrieve the partial representation of a resource with the attributes named resourceID.



Example

tm**forum** TRAINING

**Retrieve the report with an « id » of 5 and populate
the representations with the failures and metrics attributes.
Note that the « id » attribute is always present.**



REQUEST

GET /api/report/5/?fields=failures,metrics

RESPONSE

200

Content-Type: application/json

```
{  
    "id": 5,  
    "metrics": [  
        {  
            "code": "123",  
            "category_id": "45",  
            "date_time": "2012-12-20 14:30:00",  
            "reference": "4321",  
            "source_id": "8f27ce50-4b00-11e2-bcf0-0800200c9a66",  
            "value": 50,  
            "metric_id": "b0dbbc50-4b00-11e2-bcf0-0800200c9a66"  
        },  
        {  
            "code": "321",  
            "category_id": "48",  
            "date_time": "2012-12-20 14:30:00",  
            "reference": "4321",  
            "source_id": "8f27ce50-4b00-11e2-bcf0-0800200c9a66",  
            "value": 50,  
            "metric_id": "c50f3e90-4b00-11e2-bcf0-0800200c9a66"  
        }  
    "failures": [  
        {  
            "failure_id": "0bec4420-4b01-11e2-bcf0-0800200c9a66",  
            "detail": "Network authentication failure",  
            "source_id": "24b2bc00-4b01-11e2-bcf0-0800200c9a66"  
        },  
        {  
            "failure_id": "3b687c50-4b01-11e2-bcf0-0800200c9a66",  
            "detail": "Network authentication failure",  
            "source_id": "462f1f40-4b01-11e2-bcf0-0800200c9a66"  
        }  
}
```

How to retrieve resources using an attribute filtering mechanism?

- The filtering is based on using name value query parameters on entity attributes
- The basic expression is a sequence of attribute assertions being ANDED to formulate a filtering expression:

```
GET ../{resourceName}/?[{attributeName}={attributeValue}&*]
```

- Match only for attribute value equality



- The complete resource representations (with all the attributes) of all the matching entities must be returned
- The returned representation of each entity must contain a field called « id» and that field be populated with the resourceId
- If the request is successful then the returned code MUST be 200
- The exceptions code must use the exception codes from
<http://www.iana.org/assignments/http-status-codes/http-status-codes.xml>
- The Content-Range header is used to indicate the presence of more elements in the collection and the current position of the elements in the overall collection



**Retrieve all Reports with date_time
greater than 2013-04-20
and health_state operational.**



REQUEST

```
GET /api/report?date_time.gt=2013-04-20&health_state =operational      or  
GET /api/report?date_time>2013-04-20&health_state = operational
```

RESPONSE

```
200  
Content-Type: application/json  
Content-Range: items 1-2/2  
[  
  {  
    "id": << 42 >>,  
    "datetime": "2012-12-20 15:00:00",  
    "health_state": "operational",  
    "metrics": [  
      {  
        "code": "123",  
        "category_id": "45",  
        ...  
        "metric_id": "b0dbbc50-4b00-11e2-bcf0-0800200c9a66"  
      },  
      ...  
    ],  
    "failures": [  
      {  
        "failure_id": "Obec4420-4b01-11e2-bcf0-0800200c9a66",  
        "detail": "Network authentication failure",  
        "source_id": "24b2bc00-4b01-11e2-bcf0-0800200c9a66"  
      },  
      ...  
    ]  
  },  
  {  
    "id": << 52 >>,  
    "datetime": "2012-12-20 15:00:00",  
    "health_state": "operational",  
    "metrics": [  
      {  
        "code": "456",  
        "category_id": "45",  
        ...  
        "metric_id": "b0dbbc50-4b00-11e2-bcf0-0800200c9a66"  
      },  
      ...  
    ],  
    "failures": [  
      {  
        "failure_id": "Obec4420-4b01-11e2-bcf0-0800200c9a66",  
        "detail": "Network authentication failure",  
        "source_id": "24b2bc00-4b01-11e2-bcf0-0800200c9a66"  
      },  
      ...  
    ]  
  }]
```

Query Resources with attribute filtering and attribute selection

TM FORUM TRAINING

Attribute filtering and attribute selection **may be combined in a single request** as per the following example.

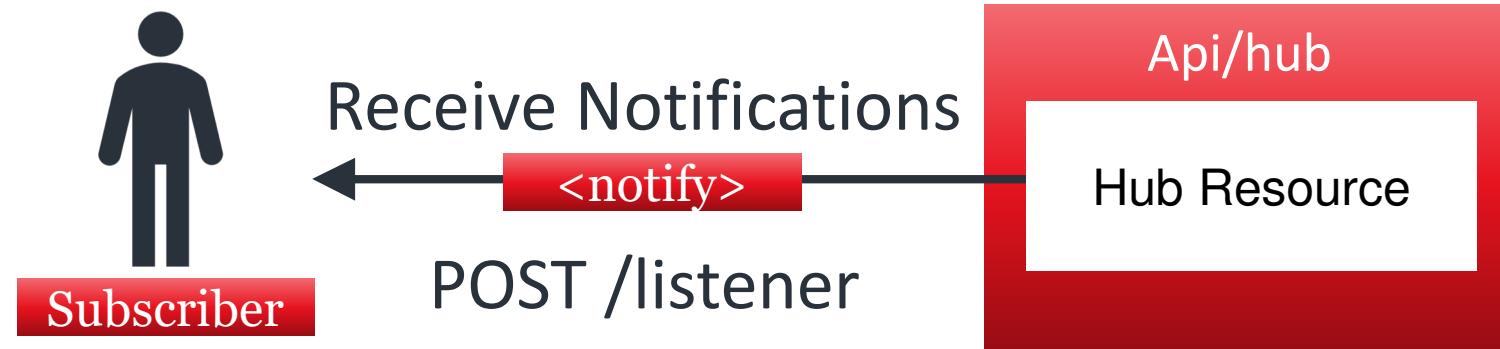
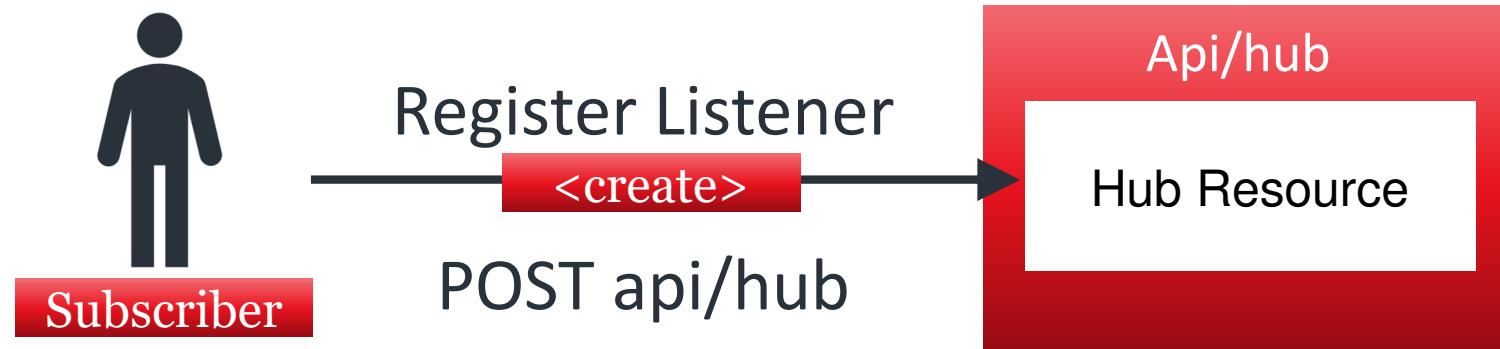
REQUEST

```
GET /api/report/fields=failures,health_state&report.date_time.gt=2013-04-20&report.state.execution=suspended
```



Notification Pattern

tmforum TRAINING



Create a Hub to receive events on the “<http://in.listener.com>”



REQUEST

POST /api/hub
Accept: application/json

{"callback": "http://in.listener.com"}

RESPONSE

201
Content-Type: application/json
Location: /api/hub/42

{"id": "42", "callback": "http://in.listener.com", "query": null}



Publishing Events

tm**forum** TRAINING

- Publishing an event is done by posting the event to the listener address.
- The structure of the event is:

```
{  
  "event": {  
    EVENT BODY  
  },  
  "eventType": "eventType"  
}
```

- Returns HTTP/1.1 status code 201



Posting Management Report

tm**forum** TRAINING

REQUEST

```
POST /client/listener
Accept: application/json

{
  "event": {
    "dateTime": null,
    "id": "42",
    "state": {
      "healthState": "UNKNOWN",
      "executionState": "ACTIVE"
    },
    "metrics": [
      {
        "code": "76e27e2b-644e-11e2-8ea1-5c260a86d1e4",
        "categoryID": null,
        "dateTime": 1367266835571,
        "reference": null,
        "sourceID": null,
        "value": "20",
        "metricID": null
        {
          "code": "7dd589b2-644e-11e2-8ea1-5c260a86d1e4",
          "categoryID": null,
          "dateTime": 1367266835573,
          "reference": null,
          "sourceID": null,
          "value": "89",
          "metricID": null
        }
      ],
      "failures": null
    },
    "eventType": "ManagementReport"
  }
}
```

Trouble Ticket API Design Patterns Walkthrough



Quiz

Why do we need REST API Design Patterns?

What are the TMF REST Design Guidelines?

Name 2 recurrent questions addressed by the TMF REST Design Guidelines?

What are TMF 630 and TMF 631?

What Uniform Operation is used to create an Entity?

What APIs are based on the TMF Design Guidelines ?



Module 5:

TM Forum Open APIs and the

Information Framework



From Logical Information Model
to API Data Models

- TM Forum's Open APIs are complementary with the Business Process and Information Frameworks.
- The respective data structure provided with every TM Forum Open API is aligned to the Information Framework
- The Information Framework is widely adopted throughout the communications industry
 - helps ensure interoperability – one of the key objectives of the TM Forum Open API program.



Resources and Entities

tm**forum** TRAINING

- Resources are **representations** of the entities managed by the application.
- Resources are acted upon by the REST API using the Uniform Contract Verbs (POST, GET, PUT, DELETE, PATCH, etc....) .
- Operations on Resources affect the state of the corresponding managed entities.
- There is a direct mapping between the managed entities and the corresponding Resources in the REST model.
- Resource identifiers represent the actual resources that a service exposes.
- A resource can be a trouble Ticket, a Logical Port , an Order, a Task etc...



SID is static, it focuses on the data

It describes the business entities

It does not describe what you can do with such business entities

SID is an information model not a data model

SID out-of-the-box is typically not implemented as is.

Some UML concepts being used cannot be directly mapped to mainstream technologies like XML, Java, etc.

TMF APIs uses SID as the basis for its data model

TMF APIs Entities are REST/JSON realizations of the SID Entities (the Data part of the Shared Information and Data).

TMF APIs adds behaviour to SID

It defines the operations that can be executed on the entities

It defines the interaction patterns Request/Response/Exceptions and Notifications

JSON Resource Model and SID Entity Types Patterns

tmFORUM TRAINING

- There is a direct mapping between the SID entity types and the corresponding JSON resources in the TMF REST API model.
- The mapping between the SID entity types and the corresponding Resource model is based on a set of patterns called the SID JSON Mapping Patterns



- The SID model define a number of relationships not all of them needs to be implemented for a particular resource within an API
- Choice of link data model must be made when mapping relationships to a REST based representation model

Hyperlinks

Link Header

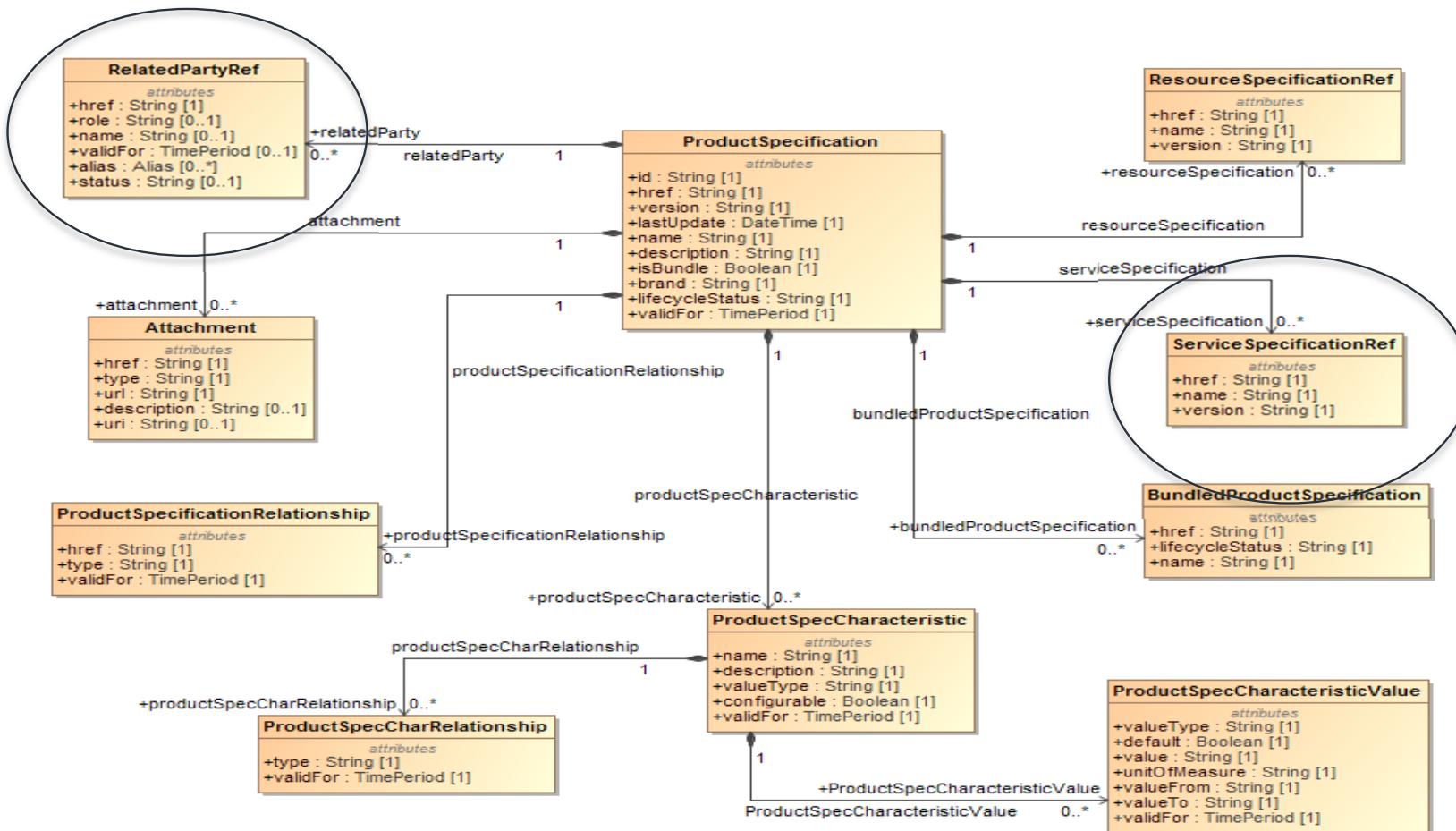
Dependent Entities



Simple Data Model with Hyperlinks... based on SID

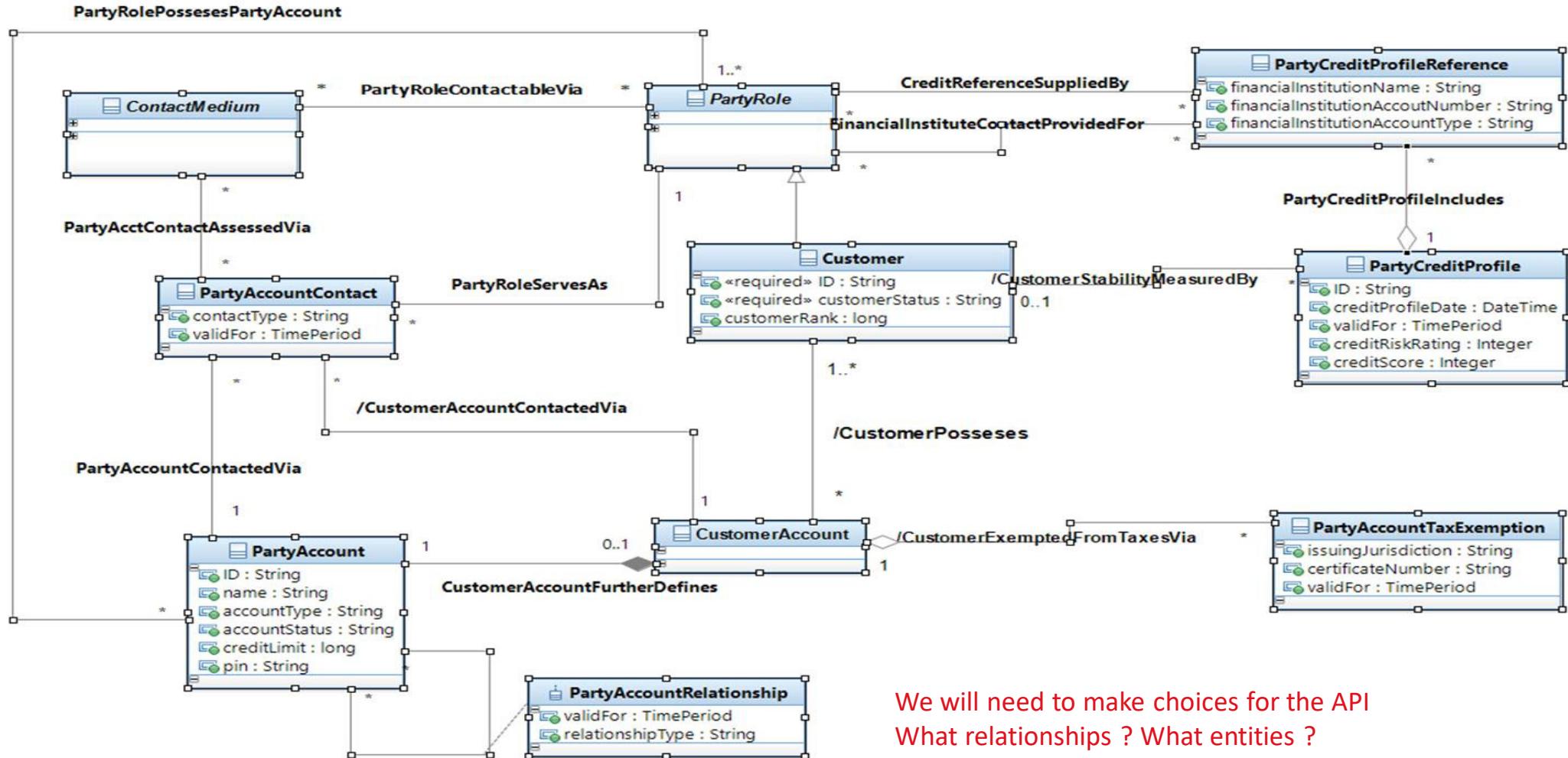
tmforum TRAINING

Product
Catalog
API



Customer SID Business Entities and the API Model

tm**forum** TRAINING



- Embed related object properties within entity representation (transform relationship to dependent entity- data type)
 - This is sometime used when there is still no resource defined for the associated entity in the API Ecosystem
- Use href or links within the JSON body (hyperlinks) with some useful filtering information
 - This is the preferred approach when the related object is treated as a resource
 - Additional properties are added to the href for filtering and quick retrieval purpose

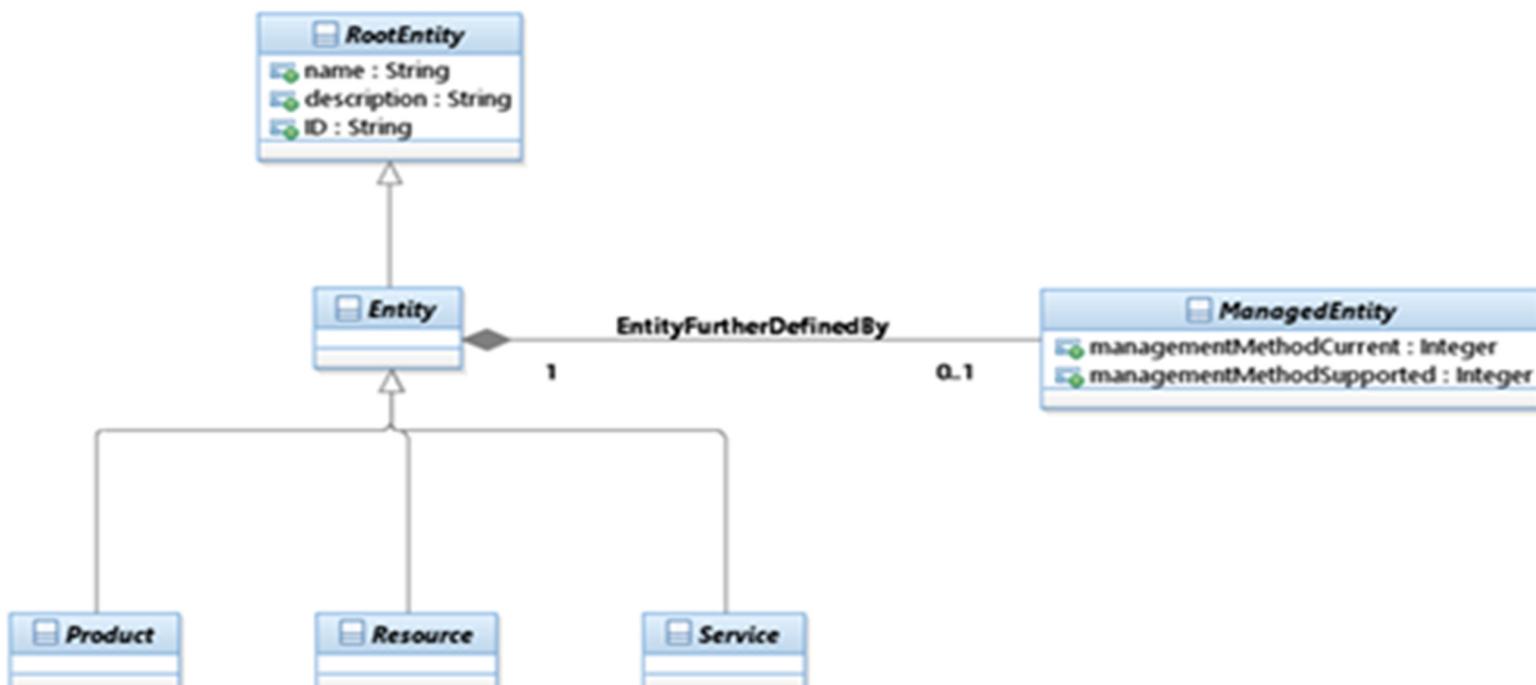


```
{  
    "id": "c1234",  
    "href": "http://serverlocation:port/customerManagement/customer/c1234",  
    "name": "DisplayName",  
    "status": "Active",  
    "description": "Description string",  
    ---dependant entity relationship---  
    "contactMedium": [  
        {  
            "type": "Email",  
            "validFor": {  
                "startDateTime": "2013-04-19T20:42:23.0Z"  
            },  
            "medium": {  
                "emailAddress": "abc@tmforum.com"  
            }  
        }  
    ],  
    ---href relationship---  
    "customerAccount": [  
        {  
            "id": "1",  
            "href": "  
http://serverlocation:port/customerManagement/customerAccount/1",  
            "name": "CustomerAccount1",  
            "description": "CustomerAccountDesc1",  
            "status": "Active"  
        }  
    ],  
}
```

Flattening Inheritance Hierarchy Pattern

tm**forum** TRAINING

- REST Resources represent SID entities.
- Some SID entities are part of inheritance hierarchies

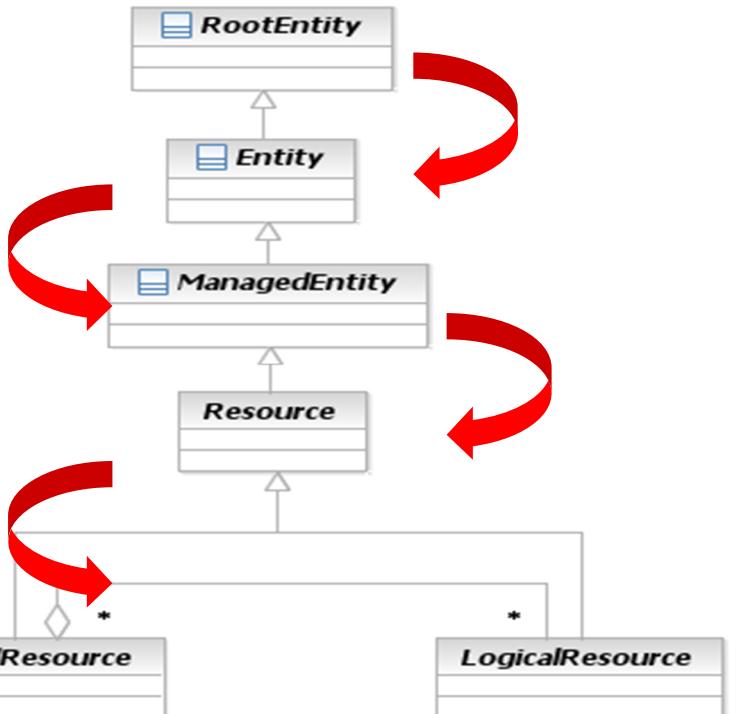


- JSON does not support generalization or inheritance (no equivalent of xsi:type at run time)
- In general only the most derived SID classes are used as Resource representations in TMF Open APIs
 - In the SID JSON representation we normally collapse the class into the direct child
 - We then expose the SID Entity as a resource with all the inherited attributes embedded into the JSON representation
- This does not mean that an API SID JSON Resource can't be extended
 - The REST API Design Patterns support the "@type" property.



Collapsing Classes

tm**forum** TRAINING



- Allow collapsing a class into either its direct parent or its direct child
- Can be recursive as show on figure:
 - In the figure case, **LogicalResource** inherits from **Resource** and **Resource** has not parent
- **If attributes and associations, handle as if present on target class**
- Only valid for direct parent or child today. Extension planned to other associations in future.



Excluding Un-needed Attributes or Associations

tm**forum** TRAINING

In the context of an API, some attributes or associations might not be needed, but might be useful in some other cases



SID/JSON Mapping Methodology

tmforum TRAINING

- Choose the Entities to be mapped to API JSON Resources
 - based on management functionality
 - within this or another API Component
- Apply Entity Collapsing Pattern
- Choose related Entities mapped to Dependent Data
 - no need to expose them as resources
 - no API exist to support them as linked entities

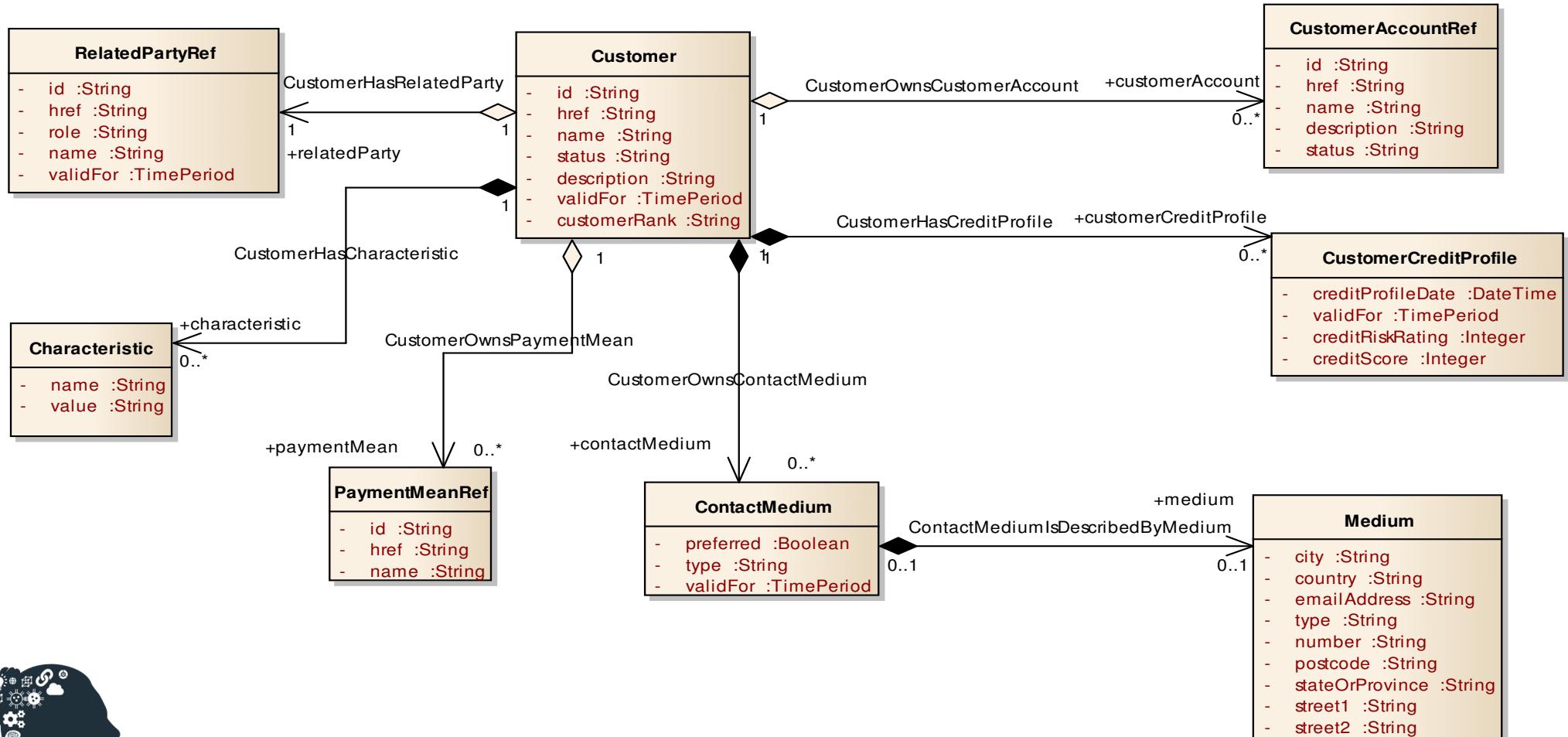


- Transform relationships to other Resource Entities into (array) of href relationships
- Transform relationships to dependent entities into array properties
- Transform Characteristics and Configurable Characteristics into JSON array of Name Value Pairs
- Transform Characteristic Specification into predefined Characteristic Specification JSON construct



Customer API Data Model

tmforum TRAINING



TM Forum 625

Open API Data Model Walkthrough



Quiz

Name 1 SID to API Data Model Mapping Pattern

Name the two Link models used in TMF Open APIs?

What level of SID inheritance is supported by the TMF Open APIs?

What is the Open API Data Model?

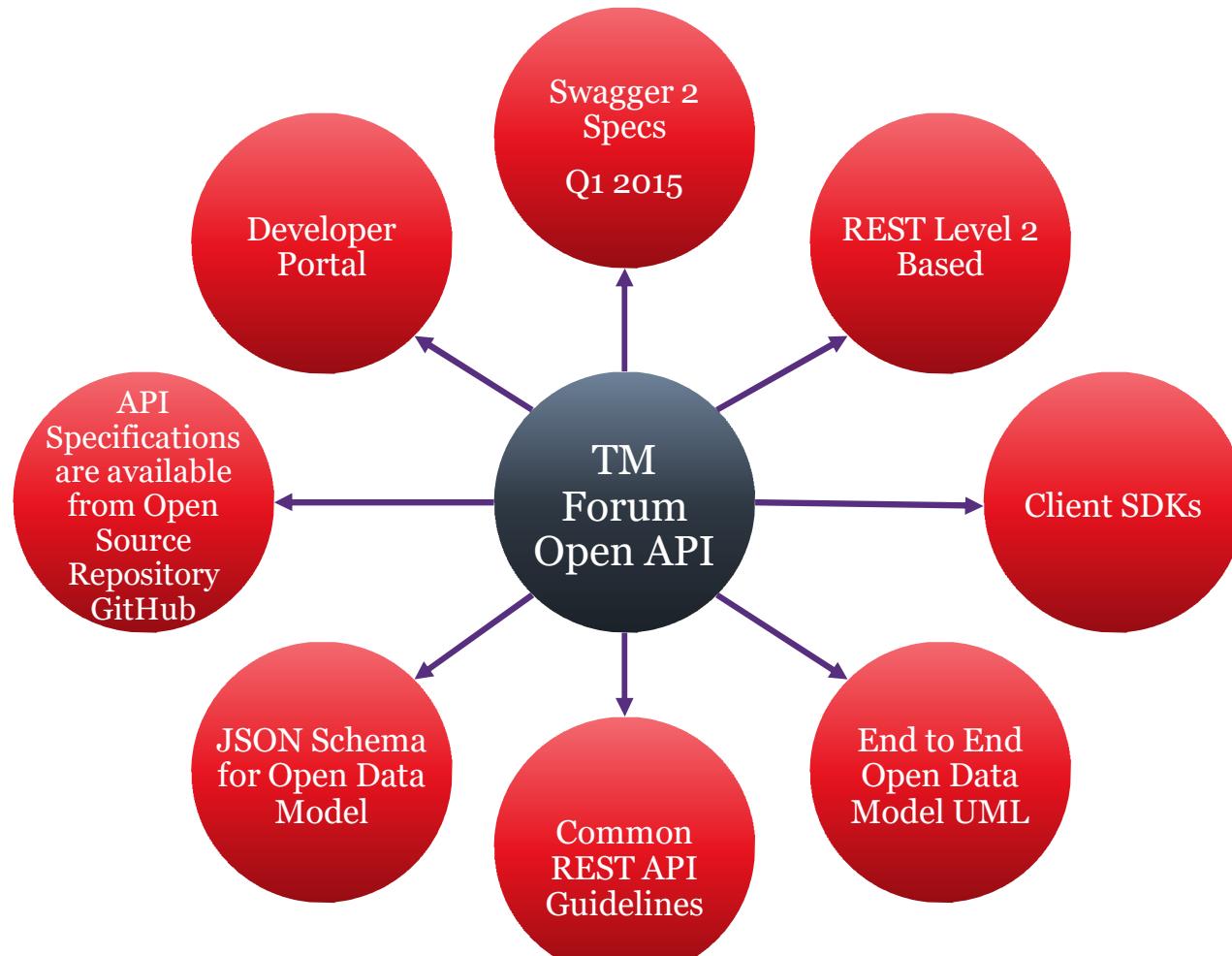


Module 6: Tooling, Sandbox, Conformance



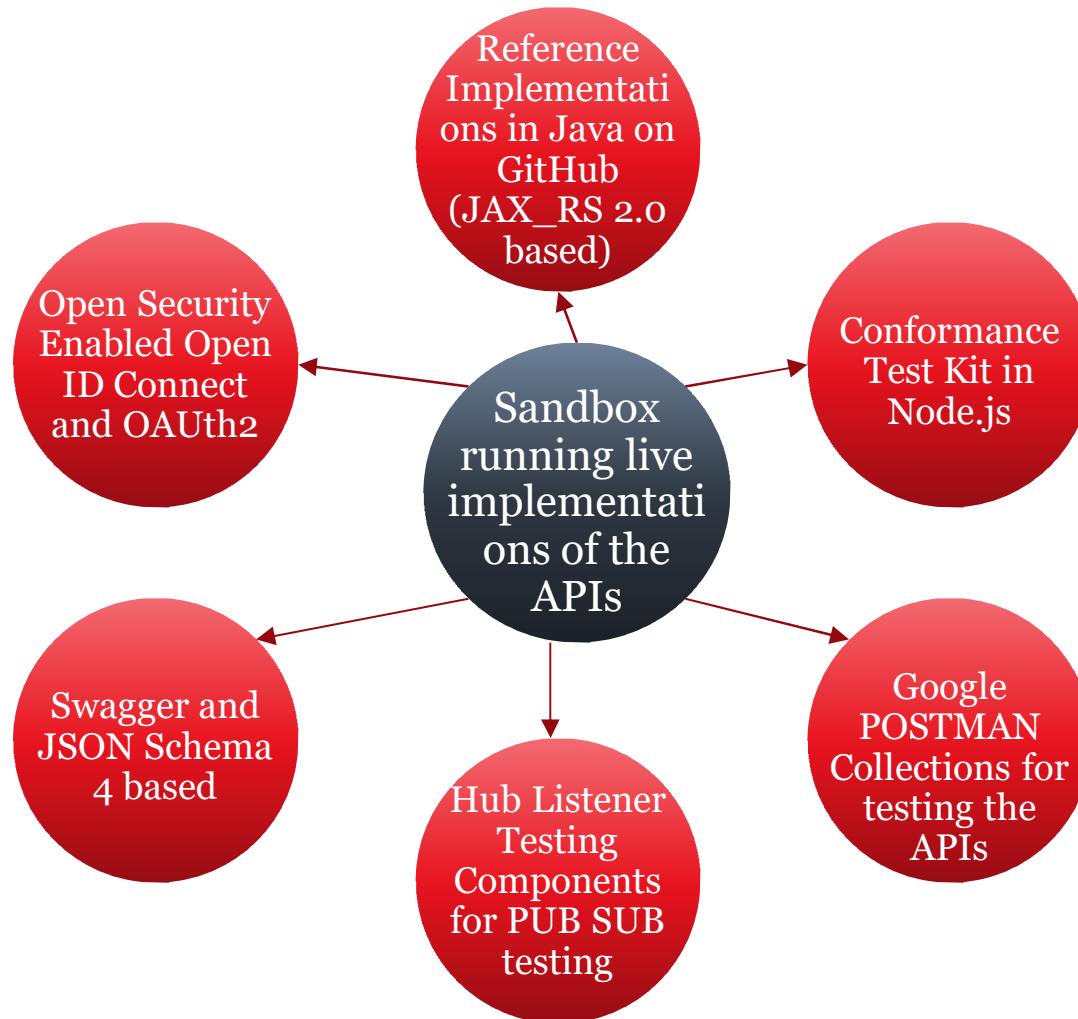
APIs are for Developers...

tmforum TRAINING



APIs Sandbox and Testing...

tmforum TRAINING



API Ecosystem Portal

tmforum TRAINING

TM Forum Ecosystem API Portal

PAGE TREE

- > Developer Quick Start Guide
- > Open API Table
 - > Trouble Ticket API REST Specification
 - > Customer Management API REST Specification
 - > Product Catalog Management API REST Specification
 - > Product Inventory Management API REST Specification
 - > Product Ordering API REST Specification
 - > Billing Management API REST Specification
 - > Party Management API REST Specification
 - > SLA Management API REST Specification
 - > Usage Management API REST Specification
 - > Performance Management API REST Specification
- > API Stories
- > Current News & Activities
- > Discussions
- > Calendar

Space tools ▾ <<

Released APIs	API Specification	Postman Collection	Swagger Swagger_UI	Reference Implementation Code	Download Specification
Trouble Ticket API Provides a standardized client interface to Trouble Ticket Management Systems for creating, tracking and managing trouble tickets among partners as a result of an issue or problem identified by a customer or another system. Examples of Trouble Ticket API clients include CRM applications, network management or fault management systems, or other trouble ticket management systems (e.g. B2B).					
Customer Management API Provides a standardized mechanism for customer and customer account management, such as creation, update, retrieval, deletion and notification of events.					
Product Catalog Management API Provides a standardized solution for rapidly adding partners' products to an existing Catalog. It brings the capability for Service Providers to directly feed partners systems with the technical description of the products they propose to them.					
Product Inventory Management API Provides standardized mechanism for product inventory management such as creation, partial or full update and retrieval of the representation of a product in the inventory. It also allows the notification of events related to product lifecycle.					

<http://projects.tmforum.org/wiki/display/API/Open+API+Table>

POSTMAN

tmforum TRAINING

The screenshot shows the Postman application interface. On the left, there is a sidebar with a search bar and tabs for 'History' and 'Collections'. The 'Collections' tab is selected, showing a list of collections: 'TMF Product Catalog API' (28 Aug at 1:03 pm, 27 requests), 'TMF Product Inventory API' (28 Aug at 12:50 pm, 12 requests), 'TMF Product Ordering API' (28 Aug at 12:50 pm, 13 requests), 'event subscriber', 'hub', 'product ordering' (selected), 'TMF SLA Management API' (28 Aug at 12:53 pm, 18 requests), 'TMF Trouble Ticket API' (28 Aug at 12:54 pm, 10 requests), 'TMF Usage Management API' (28 Aug at 12:54 pm, 14 requests), and 'TestCollection' (28 Aug at 1:42 pm, 2 requests). The 'product ordering' collection is expanded, showing methods: GET /productOrder, GET /productOrder attribute selection, GET /productOrder filtering, GET /productOrder/:id, POST /productOrder, and PATCH /productOrder/:id. The 'POST /productOrder' item is highlighted with a teal background.

The main area displays the 'Builder' tab for the '/productOrder' endpoint. The URL is '/productOrder' with the description 'create a product order'. The method is set to 'POST' with the path parameter '{[productOrderingApi]}/productOrder'. Below this, there are tabs for 'Authorization', 'Headers (2)', 'Body', 'Pre-request script', and 'Tests'. The 'Body' tab is selected, showing the raw JSON payload:

```
1  {
2    "externalId": "2",
3    "state": "Acknowledged",
4    "priority": "4",
5    "description": "reference implementation description",
6    "category": "uncategorized",
7    "requestedStartDate": "2015-04-01T16:42:23.0Z",
8    "requestedCompletionDate": "2015-04-03T16:42:23.0Z",
9    "notificationContact": "Mr. Brun",
10   "note": [
11     {
12       "text": "A free text detailing the note"
13     }
14   ],
15   "relatedParty": [
16     {
17       "role": "customer",
18       "id": "345221",
19       "href": "http://serverlocation:port/partyManagement/customer/345221",
20       "name": "John Doe"
21     },
22     {
23       "role": "agent"
24     }
25   ]
}
```

Below the body, the response status is shown as 'Status 201 Created Time 222 ms'. The response body is displayed in 'Pretty' format:

```
1  {
2    "id": 336,
3    "href": "http://env-0693795.jelastic.servint.net/DSProductOrdering/api/productOrdering/v2/productOrder/336",
4    "externalId": "2",
5    "priority": "4"
```

Buttons for 'Params', 'Send', and a copy icon are visible above the response body. A 'Save response' button is located at the bottom right of the response panel.

Swagger is a powerful open source framework backed by a large ecosystem of tools that helps you design, build, document, and consume your RESTful APIs.

- [Swagger Doc](#)
- [Swagger Specification](#)
- [Swagger Hub](#)



The equivalent of XML Schema for JSON
[JSON Schema Documentation](#)



Swagger and JSON Schema V4

tm**forum** TRAINING

The screenshot shows the Swaggerhub interface for the tmforum ActivationandConfigurationAPI / 1.0. On the left, there is an 'Editor' tab containing the raw Swagger JSON code, and an 'Interactive API Docs' tab which is currently active. The right side displays the API documentation for the POST /service endpoint.

POST /service

Summary
serviceCreate

Parameters

Name	Located in	Description	Required	Schema
service	body	serviceSpecification:	Yes	<pre>▼ service { id: integer href: string name: string description: string type: string version: string state: string serviceCharacteristic: ► Services { } serviceAdministration: ► [] ... }</pre>

Raw JSON (Editor Tab)

```
+ 1 ▾ [  
+ 2   "swagger": "2.0",  
+ 3   "info": {  
+ 4     "description": "",  
+ 5     "version": "1.0",  
+ 6     "title": "API Activation and Configuration"  
+ 7   },  
+ 8   "host": "env-0693795.jelastic.servint.net",  
+ 9   "basePath": "/DSEntityProvisioning/api/ActivationAndConfiguration/v2",  
+ 10  "schemes": ["http"],  
+ 11  "consumes": ["application/json"],  
+ 12  "produces": ["application/json"],  
+ 13  "paths": {  
+ 14    "/service": {  
+ 15      "post": {  
+ 16        "tags": ["service"],  
+ 17        "operationId": "serviceCreate",  
+ 18        "summary": "serviceCreate",  
+ 19        "description": "",  
+ 20        "deprecated": false,  
+ 21        "parameters": [{  
+ 22          "name": "service",  
+ 23          "in": "body",  
+ 24          "required": true,  
+ 25          "schema": {"$ref": "#/definitions/service"}  
+ 26        }],  
+ 27        "responses": {"201": {  
+ 28          "description": "service",  
+ 29          "schema": {"$ref": "#/definitions/service"}  
+ 30        }}  
+ 31      }  
+ 32    }  
+ 33  }  
+ 34 }
```

The screenshot shows the SwaggerHub interface for the tmforum Appointment API. The top navigation bar includes 'APIs ▾', 'Show All', 'GO', and a dropdown for 'tmforum'. Below the header, the URL is '/apis / tmforum / Appointment / 1.0 ▾'. The interface has tabs for 'Editor' and 'Interactive API Docs', with 'Interactive API Docs' selected. A 'Read Only' message is displayed over the code editor area. The main content area is titled 'API Appointment' and 'Version 1.0'. It lists a single endpoint: 'POST /appointment'. The 'Summary' section describes the operation as creating an appointment with all its characteristics. The 'Description' section provides a detailed explanation. The 'Parameters' section shows a table with one row for the 'Appointment' schema, which includes fields for 'alarm' (boolean), 'attachment' (array), 'category' (string), and 'endDate' (string). The bottom right corner features a 'Help make SwaggerHub better' button with a user profile picture.

```

1  swagger: '2.0'
2  info:
3    description: ''
4    version: '1.0'
5    title: API Appointment
6    host: api.orange.com
7    basePath: /
8    schemes:
9      - http
10   consumes:
11     - application/json
12   produces:
13     - application/json
14   paths:
15     /appointment:
16       post:
17         tags:
18           - appointment
19         operationId: appointmentCreate
20         summary: create
21         description: 'After checking free slots, this operation is used to create an appointment with all its characteristics.'
22         deprecated: false
23         parameters:
24           - name: appointment
25             in: body
26             required: true
27             schema:
28               $ref: '#/definitions/Appointment'
29         responses:
30           '201':
31             description: Success
32             schema:
33               $ref: '#/definitions/Appointment'
34           '400':
35             description: bad_request
36             schema:
37               $ref: '#/definitions/Error'
38         get:
39           tags:
40             - appointment
41           operationId: appointmentList
42           summary: find
43           description: "This operation is used to retrieve appointment information using"

```

The screenshot shows the Swaggerhub interface for the ActivationandConfigurationAPI / 1.0. The top navigation bar includes a logo, a search bar with dropdowns for 'APIs' and 'Show All', a 'GO' button, and a 'tmforum' dropdown. Below the header, the URL is / apis / tmforum / ActivationandConfigurationAPI / 1.0. A toolbar provides options for 'PUBLIC', 'UNPUBLISHED', '...', 'Download', and 'Settings'. The main content area has tabs for 'Editor' and 'Interactive API Docs' (which is selected). On the right, a status bar shows 'Last modified: 01:47:18 am May 3, 2016' with a 'VALID' status and a 'Save' button. The API documentation lists operations under two sections: '/resource' and 'service'.

/resource

POST	/resource	resourceCreate
DELETE	/resource/{resourceId}	resourceDelete
GET	/resource/{resourceId}	resourceGet
PATCH	/resource/{resourceId}	resourcePatch

service

GET	/service	serviceFind
POST	/service	serviceCreate
DELETE	/service/{ServiceId}	serviceDelete
GET	/service/{ServiceId}	serviceServiceGet
PATCH	/service/{ServiceId}	ServicePatch

[BASE URL: /DSEntityProvisioning/api/ActivationAndConfiguration/v2 , API VERSION: 1.0]

What is API Certification ?

tm**forum** TRAINING

Certification is the process by which an implementation is confirmed to support the standard definition of an API

TM Forum provides the tools required to certify implementations against API definitions

Certification is only against Mandatory features of an API



Main Artifacts

tm**forum** TRAINING

API Conformance
Profile document

Describe what is tested Mandatory attributes and Operations

CTK script

Provided by TMF to be executed against certified implementation of the API

Cloud based
Reference
Implementation
of the API

Provided by TMF as implementation test bed
Is an example of a certified API



What is tested by a CTK ?

1. Implementation of all the mandatory operations
2. Support of the core data model of the API presence and proper constructs for entities in the data model
3. Support of mandatory filters and attribute selection constructs as defined in the API specification
4. Support of all mandatory notifications
5. Support of the core event data model of the API
6. Mandatory relationships between entities (resources) **defined** and **managed** by the API are present and operation(href leads to real resources within the API)
7. Mandatory relationships between entities (resources) not **defined** and **managed** by the API are present



What is required to be conformant ?

tm**forum** TRAINING

An API must pass all the CTK tests in order to be approved as a valid implementation of the TMF specification

An API must have implemented all the CORE or MANDATORY features to be conformant.

If a mandatory feature is missing then the API is eliminated and out of consideration for conformance



Trouble Ticket API Conformance Profile Walkthrough



Trouble Ticket API CTK Example



Property of TM Forum.

For official TM Forum Training participant personal use/reference only.

Quiz

Name elements of the Open API Developer Ecosystem Sandbox and Testing environment?

What is the TMF Open API Sandbox?

What is the API Certification about?

Name 2 elements that are tested via the CTK?

Can an API be certified if it fails one CTK test?



Module 7:

Introduction to TM Forum MicroServices



Microservice architecture is a design pattern to build software system composed of multiple reusable services that interacts with each other to provide a business capability.

Each MicroService is

Isolated

Autonomous

Context-bound



“

Isolation is at the heart of microservices. A microservice is meant to be architected, created, deployed, maintained and retired without affecting any other microservice. You can't do any, let alone all, of that without good isolation

Donald Belcham

”

Isolation helps in building and maintaining a single capability without impacting other functions.

Isolation principle enables:

- Design, Build, Test and Release isolation - Enables continuous delivery, reduced time to market and time to value as change is isolated to a specific business function
- Database isolation - Eliminates integration at database level. Any changes in database affects only one microservice and thus reduces surface area for testing, validation and deployment
- Failure isolation - Contain propagation of failure and prevent cascading impact to whole system

- Microservices are autonomous, self-governing and controls their own destiny.
- Consumers cannot impose obligation (Command & Control) on provider.
- An autonomous provider service can only promise its own behavior by publishing its interface (API)
- Consumer can choose to accept or reject the promise offered by provider but cannot direct it to behave in a particular way.



Data Encapsulation

tm**forum** TRAINING

Microservice
encapsulates the data
(or state) and behavior
as a single unit.

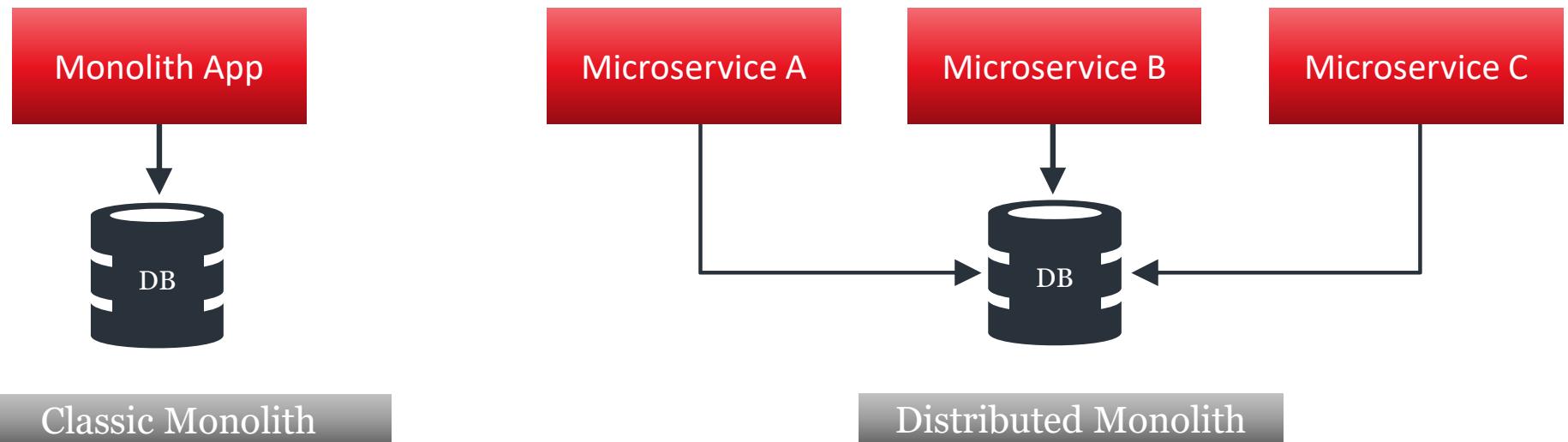
Persistent data for
each service must be
private.

Consumer can access
data only through its
published interfaces
or API.



Distributed Monolith Anti Pattern

tm**forum** TRAINING

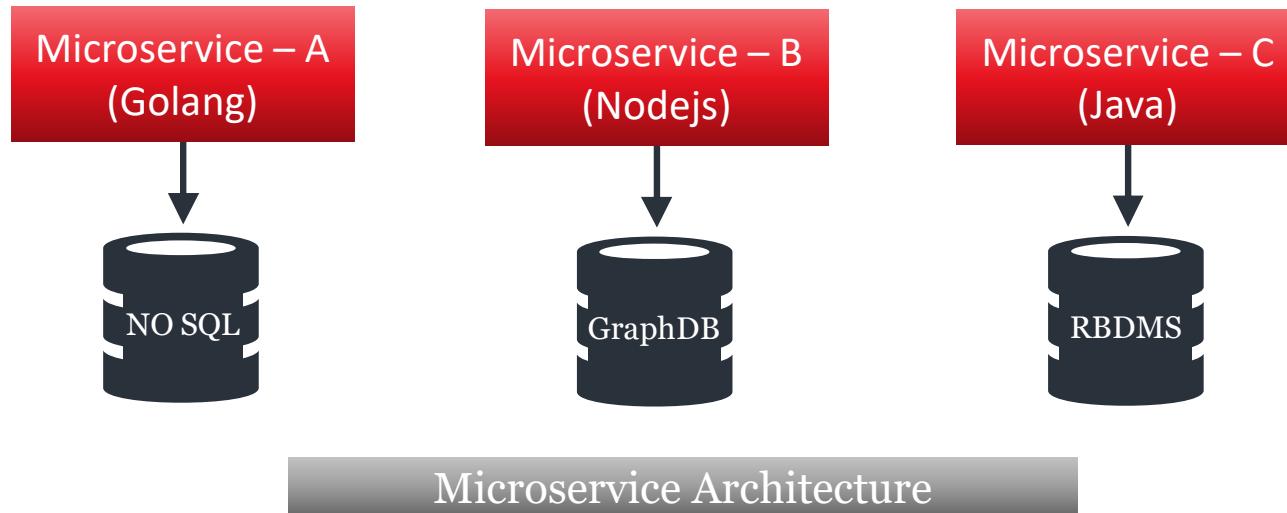


Distributed Monolith is a major anti-pattern where bunch of so called independent services shares a common database for persistent state.



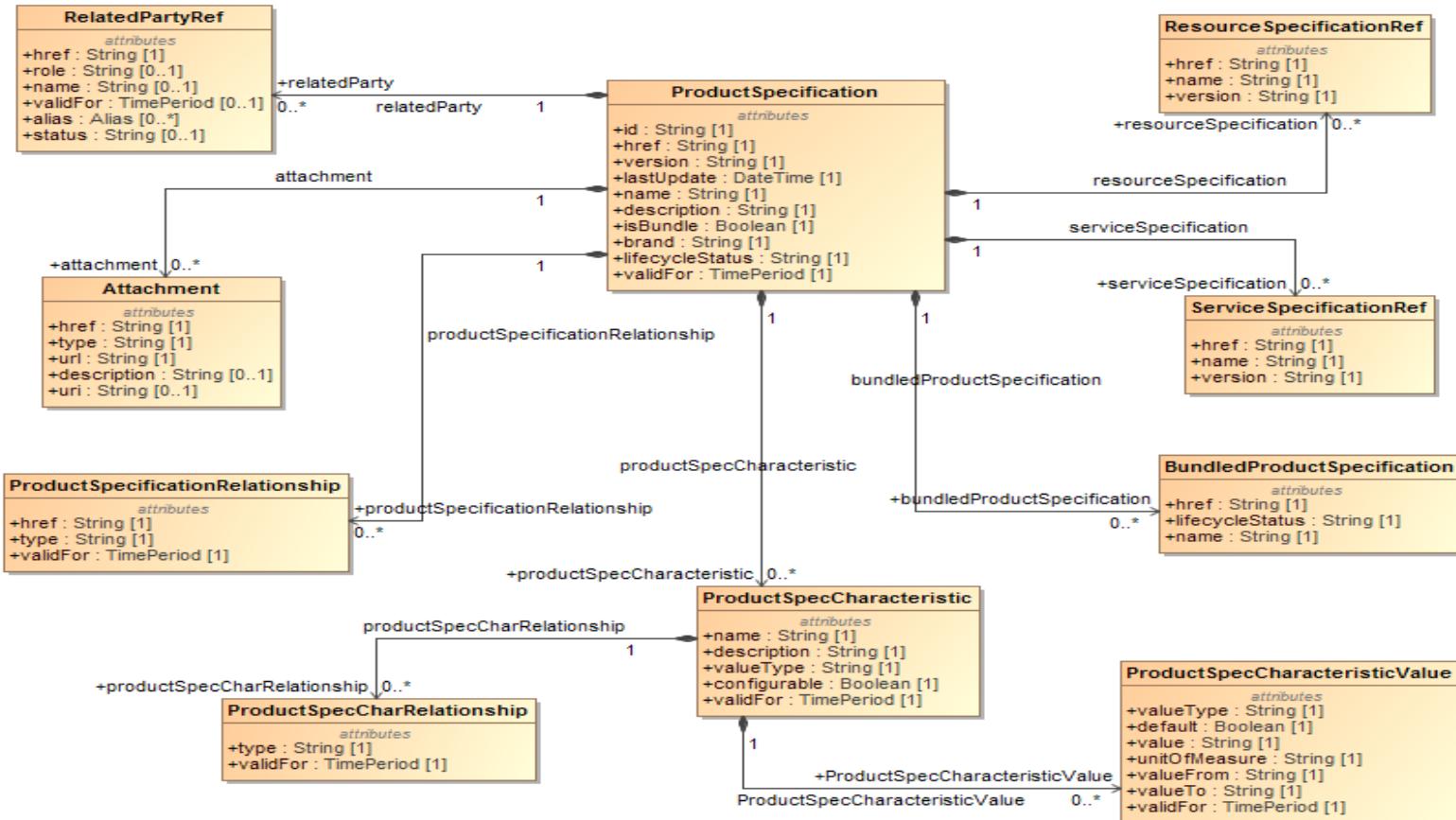
Persistency Isolation

tm**forum** TRAINING



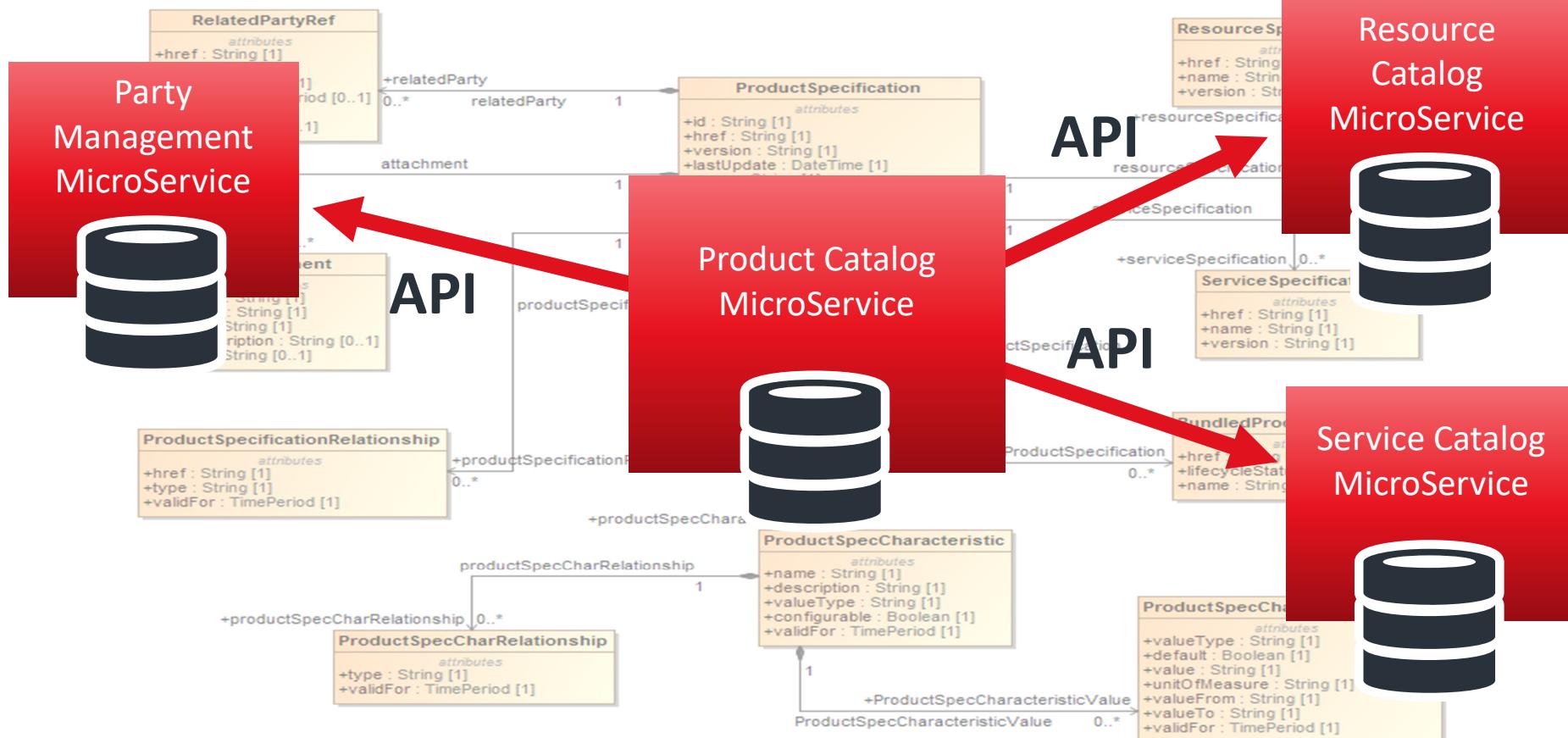
Domain Driven Open API Micro Services

tmforum TRAINING



Domain Driven Open API Micro Services

tmforum TRAINING



Expose one MicroService per API managed Resource.

Quiz

tm**forum** TRAINING

What is a Micro Service Architecture?

Name one fundamental characteristic of a MicroService?

Is building MicroServices on top of an existing centralized database a good practice ?

Name one way of realizing Open APIs with MicroServices ?



Review & Wrap-up



Property of TM Forum.

For official TM Forum Training participant personal use/reference only.

Thank You!

