ORACLE®
Financial Services

Accelerate Innovation With API Banking

Fast track Open Banking with a packaged API solution

FEATURING RESEARCH FROM FORRESTER

Keep API Strategy On Track With An API Taxonomy

Banking as a business has always adapted to the prevailing nature of transactions. Beginning with the barter system to the introduction of paper currency, plastic money, electronic wire transfers and mobile payments, banking has integral part of these evolutions. Another radical era in banking is now emerging, consumer technologies have begun to talk to each other giving customers a near invisible bank. API technology with its networking capability is enabling this connectivity between digital assets and can be called the nervous system of today's digital environment.

OPEN API PLATFORMS: CONNECTING BANKS AND FINTECH'S TO PROVIDE CUSTOMERS THE BEST OF BOTH WORLDS

The digital banking customer expects privacy, security, real-time payments and an intuitive experience across any device. Fintech's and challenger banks tend to perform better on these parameters while established banks get weighed down by legacy systems. While the challengers are nimble and disruptive established players have trust on their side, gained over decades of service. Banks also have the experience and expertise of complying with regulations, they also own banking licenses and most importantly have large volumes of historic customer transaction data. Collaboration between challengers and incumbents is now a necessity, in order to meet rising customer demands. This collaboration is underway. Opening up banking APIs to a challenger's service offerings allows banks to deliver greater value to their customers. Consumers now have the ability to conduct transactions with minimum effort via wearables, contactless devices, voice interfaces and biometrics.

WHY IS A WELL-DEFINED API TAXONOMY REQUIRED TO KEEP A BANK'S API STRATEGY ON TRACK?

A well thought out API ecosystem shortens time to market enabling businesses to react faster to market dynamics. In an API ecosystem an API strategy is a must, how does a bank choose which challenger to collaborate with? Does the bank expose its systems via Open API or tailor made Private API? How do banks track usage of their APIs? Are the APIs feeding front-end devices or corporate ERP systems? Will the APIs work with different Operating systems? Can existing SOAP services exposed from a legacy core product processor provide intelligent services which adapt to the current banking environment which utilizes REST and JSON? How does a Bank manage APIs for different API business models like license, freemium, subscription and pay-as-you-go?

Banking APIs connected to an auto trading portal or real estate portal can enable end customers understand their credit worthiness and shortlist vehicles or property on the go instead of calling their bank or checking the bank website. This kind of convenience needs be made available through APIs for a seamless banking experience which ultimately results in higher revenue realization for the bank and its partners.

IN THIS DOCUMENT

API Taxonomy is an important part of an API ecosystem which is often neglected and makes managing the API ecosystem a difficult task. It needs meticulous planning and foresight. Furthermore, banking as a business is highly regulated and must have uncompromised security standards. These factors make API design and categorization a critical task. Banks also need to comply with regulations like PSD2, Open API standards like the Berlin group. An API taxonomy is integral to building mature APIs, maintaining robust security and making systems available for audit at all times

Banking ecosystems are complex in nature and span across back-office, mid-office and self-service channel solutions. These systems communicate with each other to provide end to end solutions as per business relevance. These assets are acquired over a period of time. This results in an accumulation of the bank's intellectual property and the customer information residing within these systems. Moving them to new generation solutions are often not practical or easy solutions. These systems generally expose legacy services or internal APIs. These APIs are consumed by the mid-office applications or self-service and assisted channel solutions.

APIs emerge from different architectural layers. From the core emerge source APIs that connect core product processors, these APIs support CRUD (Create, Read, Update and Delete) operations. Processors from the engagement layer allow for informational APIs which define process and tasks, Engagement APIs which orchestrate functions like pricing, originations, billing etc., The Experience layer comprises, Experiential APIs and Assisted channel APIs, which provide information to customer facing channels like mobiles, desktops, wearables, IoT devices, voice interfaces and digital personal assistants. The experience channel also includes non-functional requirement (NFR) APIs which define system features such as reliability, security, maintenance, scalability, and usability.

Persona based experience APIs, engagement APIs and non-functional requirements like customer entitlements, limits etc. are essential for developing new experiences which address the needs of reliability, security, maintenance, scalability, and usability.

A well-defined API Taxonomy across different architectural layers provides a sound framework for designing APIs within the banking ecosystem.

Figure 1. Oracle Banking APIs offers a structured approach to managing a banking API ecosystem

A PACKAGED API SOLUTION IS THE BOOST A BANK'S API ECOSYSTEM NEEDS

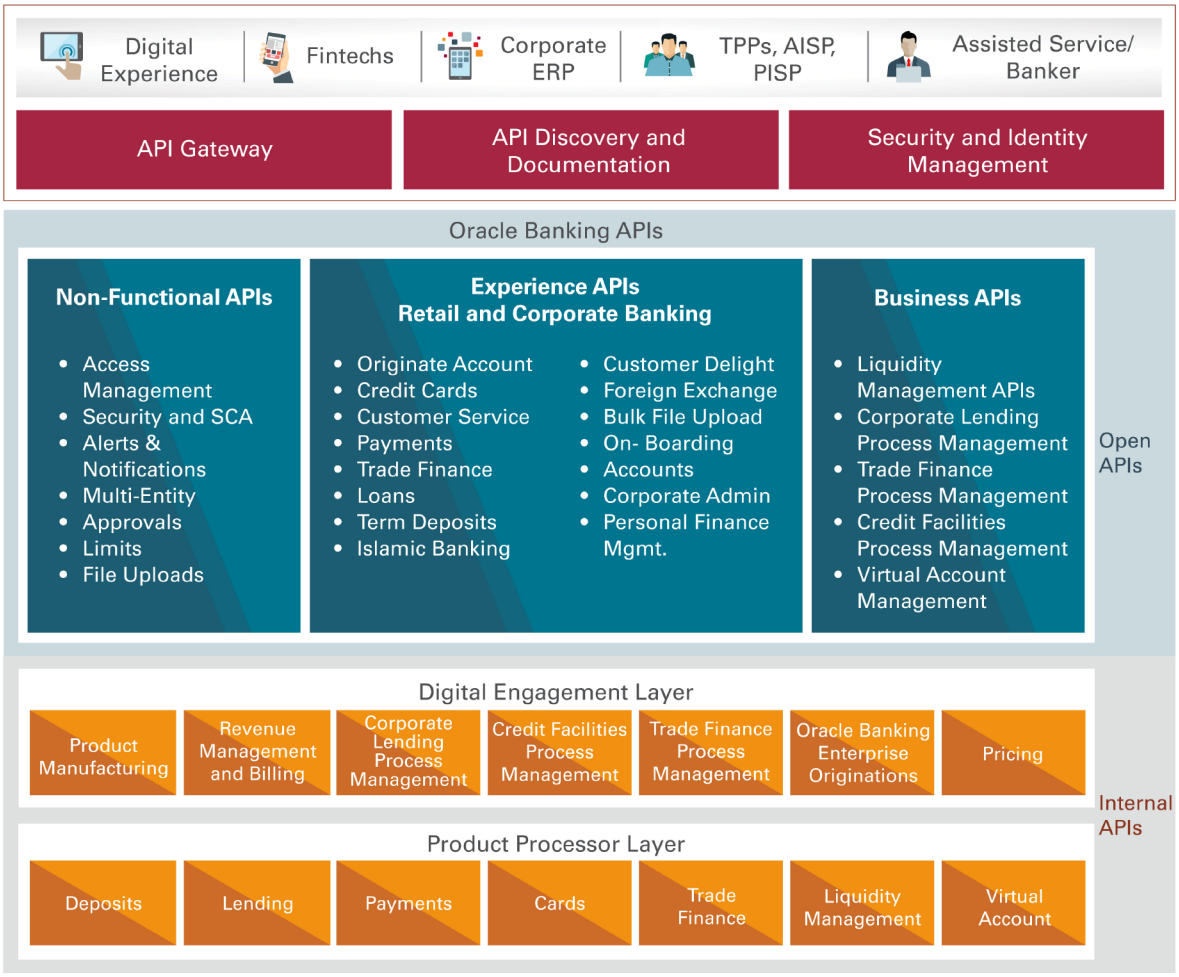
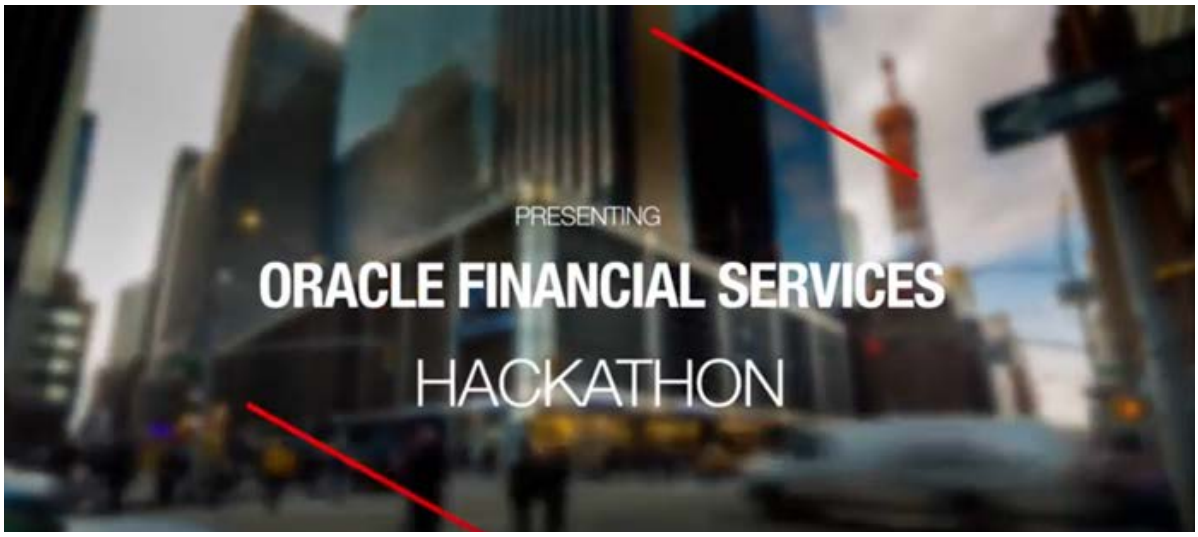


Figure 2. Oracle Banking APIs reference architecture illustrates how the packaged API solution integrates with a bank's existing IT landscape.

In the today's digital era, banks need a solution that connects their API gateway and legacy banking applications, helping them grow and maintain their API portfolio. The solution must integrate seamlessly with the existing technology landscape of the bank without disrupting the day to day functions of the bank. Employing a packaged API solution like Oracle Banking APIs aids in industrialization of APIs enabling faster revenue realization, smoother version control, change management and easy referenceability with an API interface which is easily discoverable on API frameworks like Swagger, Apiary etc. The API framework includes more than 1500 RESTful APIs Ready for Consumption.

The success of a bank's API strategy depends how easy it is for third party developers and fintechs to access their banking platform. Oracle Banking APIs provides industry standard RESTful APIs which have been put to the test at the Oracle Fintech Ecosystem. Oracle empowers banks to accelerate

their innovation cycles by investing in industry scale security, API management capabilities, functional robustness and privacy technology which is necessary to integrate with Fintechs. These investments enables banks to hand pick mature solutions to enrich their product portfolio, along with proof points of possible synergies utilizing Oracle Banking APIs. Banks can readily connect with startups and Fintech's offering solutions pre-integrated to the Oracle API library.



https://www.youtube.com/watch?v=ArsHrHOYvus

Keep API Strategy On Track With An API Taxonomy

Taxonomy And Portfolio Management Are Critical Foundations For API Success

by Randy Heffner

May 16, 2017

Why Read This Report

Application programming interfaces (APIs) underpin digital transformation by enabling Agile solutions and new business strategies. As application development and delivery (AD&D) pros create many APIs, they must: 1) organize and classify of them and 2) ensure good design of many different types of APIs. An API taxonomy addresses these concerns and more as part of one's API governance and management disciplines. This report provides AD&D pros with principles and structures for establishing and refining an API taxonomy.

Key Takeaways

API Taxonomy Is Critical, But Often Missed

When Forrester examines organizations' API strategies, whether via a formal review project or informal inquiry conversations, API taxonomy, API portfolio management, and API business strategy are the most important elements that are often missing.

Key Principles And Structures Underpin A Robust API Taxonomy

An API taxonomy helps teams understand different types of APIs, how to design them, and how they relate to one another. Important aspects in defining an API taxonomy include API usage scenarios, API purpose and scope, architecture layer, and sometimes business domain.

Avoid The Mistake Of Confusing API Taxonomy With API Implementation Patterns

An API's interface, not its implementation, distinguishes its place in an API taxonomy. Thus, it is a mistake to include characteristics such as atomic-versus-composite APIs in an API taxonomy. Instead, these belong to a related API governance discipline: API implementation patterns.

Keep API Strategy On Track With An API Taxonomy

Taxonomy And Portfolio Management Are Critical Foundations For API Success

by Randy Heffner
with Christopher Mines, Amy Homan, and Andrew Reese

May 16, 2017

Table Of Contents

Related Research Documents

A Developer's Guide To Forrester's Strategies For API Success

Establish Your API Design Strategy

How APIs Reframe Business Strategy

FORRESTER®

Forrester Research, Inc., 60 Acorn Park Drive, Cambridge, MA 02140 USA
+1 617-613-6000 | Fax: +1 617-613-5000 | forrester.com

API Taxonomy Is One Of Three Often-Missed Aspects Of API Strategy

APIs are critical for digital business. They create agility within an organization's solution architecture, which in turn enables faster delivery of business change. Even more important, APIs enable new angles into business strategy.[1] But executing on an API strategy is hard work, complicated by the fact that organizations must create many APIs of many different types. API taxonomy provides an important framework for guiding each API's design according to its type. When Forrester interacts with AD&D pros, architects, and their colleagues to examine their organizations' API strategies, whether via a formal review project or informal inquiry conversations, API taxonomy — the focus of this report — is one of three major aspects we most frequently find missing:

› **API business strategy.** We often hear phrases indicating that a client's strategy should be business-driven or closely connected to business requirements, but API business strategy is something altogether different. It's not about doing projects for the business; rather it begins with questions concerning business models, distribution channels, ecosystem leverage, market reach, customer empowerment, and the like.[2] APIs enter in as a second order concern for how to foster digital connections that answer these questions. API business strategy doesn't *support* digital transformation, it *is* digital transformation.

API business strategy doesn't *support* digital transformation, it *is* digital transformation.

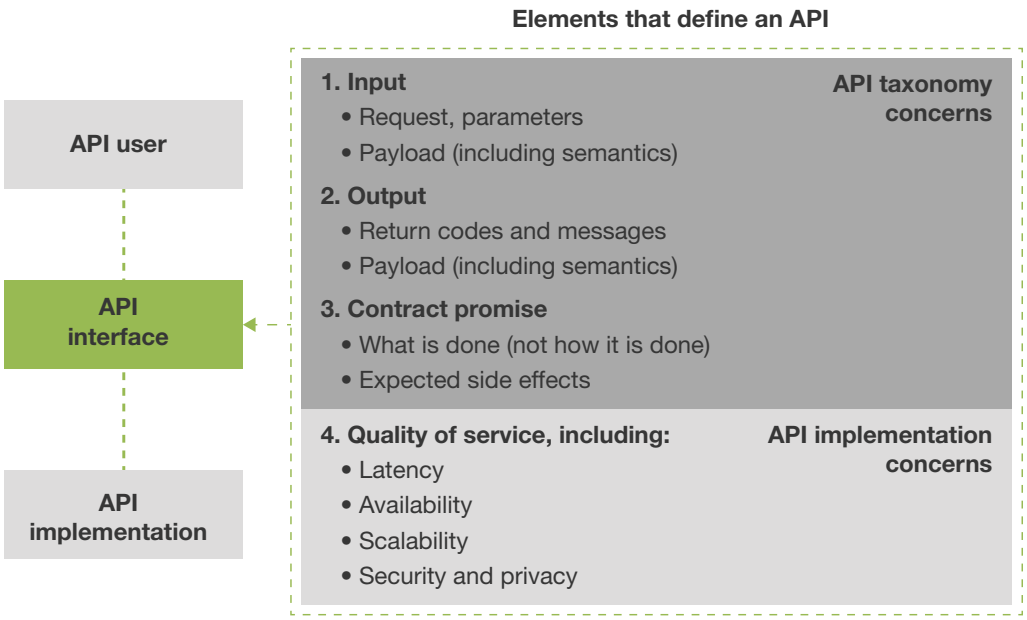
› **API portfolio management.** We often see clients let their collections of APIs grow haphazardly with each project's individual needs. After project completion, they may catalog APIs into a library, hoping that developers will use them on other projects. All too commonly, this approach fails because developers don't look for APIs to reuse, they don't find them when they do look, or the APIs don't work for them when they find them. In contrast to the haphazard approach, the best practice is API portfolio management, wherein developer-architect collaboration guides API design and evolution using lightweight definitions of target API portfolios. As projects get started, teams use these portfolios to identify existing APIs to use and candidate APIs to build. Instead of leaving it to chance, the collaboration builds reuse and coherent API evolution into project and product plans.[3]

› **API taxonomy.** API portfolio management guides the coherent evolution of an organization's collection of APIs, but teams also need guidance for structuring API portfolios and designing individual APIs. For this, API best practice begins by recognizing that APIs have varying usage scenarios with different requirements for interface design, quality of service, implementation patterns, and governance. In other words, there are many types of APIs. Thus, effective execution of API strategy requires a taxonomy to define and organize the different types of API an enterprise uses. Various API types, in turn, have different needs and best practices for applying API portfolio management.

FIRST THINGS FIRST: ESTABLISH CLEAR GUIDANCE FOR API DEFINITION

To understand the different types of APIs within a taxonomy, it is important to understand the four major elements of an API definition (see Figure 1). Of these, three (input, output, and contract promise) primarily concern API taxonomy definition. The fourth (quality of service, AKA nonfunctional requirements) primarily concerns API implementation but may also factor into API taxonomy definition. Crucially, the design of an API's interface should be separate from the design of its implementation. The implementation may take on any form that delivers the functionality and quality of service the interface design promises.

FIGURE 1 Definition Of An API Taxonomy Considers Three Of The Four Elements That Comprise An API Definition



The implementation of an API is distinct from its interface. Treat the two, along with their design and governance concerns, separately. Furthermore, the API implementation includes any and all infrastructure, applications, and data necessary to fulfill what the API interface definition promises.

Start With Key Principles And Practices To Structure An API Taxonomy

The most important best practice with API taxonomy is to have one. At minimum, it provides common language for collaboration on API design and implementation. At best, it provides a foundation for business agility (through faster realization of API-based digital business services), efficient development (through clearer guidance on API design and layering), and robust operations (by tying best practice implementation patterns to the API taxonomy). An API taxonomy will vary from one organization to another depending on enterprise maturity, developer culture, and architecture practices. Because a taxonomy will evolve as an API strategy matures, it's not worth overthinking it at the start.

That said, best practice principles and structures should guide development of an API taxonomy. The most important single practice to get right is basing an API's taxonomy solely on concerns that affect its interface design. A separate aspect of API strategy, implementation patterns, addresses questions about the program code and infrastructure behind that interface. The key question in designing a taxonomy is: "What differentiates one taxonomy item from another?" A taxonomy may have multiple levels or dimensions, so "item" may refer to a layer or category within the taxonomy or to individual classifications within a layer. For structuring an API taxonomy, the important factors are:

- **Broad usage scenarios.** Forrester identifies four broad API categories based on primary usage scenarios: open web, B2B, internal, and product integration.[4] Because each has different user audiences, business dynamics, and ecosystem considerations, an API taxonomy should treat them differently.
- **Scope.** An API that retrieves data from an individual application (e.g., customer data in SAP *or* Salesforce) has a smaller scope than an API that retrieves that same type of data across multiple applications where the data may reside (e.g., combined customer data from SAP *and* Salesforce). Forrester refers to the former as an application API. If an API of the latter type has a large enough scope to retrieve any customer no matter where it may be stored across all of an organization's applications, Forrester refers to it as a business API, since it delivers an enterprise business view of the data and frees API users to focus on the organization's business rather than its mess of duplicate and disconnected legacy applications.

A business API frees users to focus on the organization's business rather than its mess of duplicate and disconnected legacy applications.

- **Processing purpose.** Although one must always be careful of letting an API's implementation bleed through into the API taxonomy, certain differences of purpose may help to guide stronger discussions on API design. For example, over and against the arguments some REST API proponents make that the purpose of all REST APIs should be solely to create, read, update, and/or delete (i.e., CRUD) data/resources/nouns (e.g., GET customer, PUT address) stands the

fact that the world of business also includes verbs/actions/transactions.[5] An API may be more comprehensible when its design reflects a specific purpose. E-Trade's API retrieves account info via a data API (e.g., "rest/accountbalance/{accountId}") but makes trades via transaction APIs (e.g., "rest/previewequityorder" and "rest/placeequityorder").

- **Architectural purpose or layer.** APIs may serve specific layers within an architecture or purposes within a layer. Business APIs provide the most strategic API layer, since they make an organization's business data and transactions readily available to any user experience (UX), business process, or ecosystem partner. Above the business API layer, the UX API layer provides touchpoint-specific APIs (e.g., for mobile apps, or even for a specific mobile OS or platform) and multitouchpoint experience APIs (e.g., to provide familiar processing across any digital experience).[6] An organization's infrastructure layer may include APIs for security, logging, general technical utilities, or other common services.
- **Domain.** At times, it may be useful to differentiate APIs by domain, such as enterprise customers versus consumers. However, to justify having them in a taxonomy, domains need distinguishing business or design considerations that apply to more than a few APIs. For example, if government scenarios require most APIs to have additional request parameters or conform to special auditing requirements, an API taxonomy might reasonably include "business" and "government" as distinct classifications. But if only a small number of government scenarios had such requirements, it would be more efficient to simply have distinct API names (e.g., "businessOrder" versus "governmentOrder").

MISTAKES TO AVOID IN FORMING AN API TAXONOMY

AD&D pros and architects may confuse a few characteristics of APIs as being distinguishing factors of an API taxonomy, but within the landscape of API guidance and governance, they instead belong to the discipline of API implementation design and patterns. For example, an API taxonomy should not specifically call out:

- **Atomic versus composite APIs.** Forrester frequently encounters API taxonomies that distinguish between atomic APIs, which implement a small function (however an organization may define "small") and composite APIs, which call multiple atomic APIs to accomplish a broader task. While it makes sense for an API strategy to define implementation patterns for atomic and composite APIs, API users do not care about such details (nor should they), so they should not be part of an API taxonomy.
- **Security and privacy requirements for external or internal use.** Although security and privacy requirements (e.g., authentication, authorization, auditing, etc.) help define an API interface, they do not typically distinguish different types in an API taxonomy. Instead, an organization can define and apply different security policies to the same API according to specific needs for different API user groups (e.g., open web API users versus B2B API users). An API implementation pattern would establish how to apply different security policies via an API gateway or other API platform element.[7]

For example, a gateway could allow selected data in responses to B2B API users while stripping it out in responses to open web API users. Nonetheless, security requirements may factor into the definition of a business domain that is part of an API taxonomy. For example, the definition of a hospital firm's medical records business API domain would include HIPAA compliance requirements.

› **System of record or legacy application APIs.** The implementation of an API is transparent to its users — as well it should be. So it does not matter whether the business logic of an API resides in a vintage application (i.e., sometimes referred to as a "system of record"), new custom code, a software-as-a-service (SaaS) application, or any other source.

Start With High-Level Samples To Define Your API Taxonomy

Evolve your organization's taxonomy along with your API strategy. Don't try to define a theoretically perfect taxonomy upfront; rather, start with a minimal taxonomy based on the scope of your organization's API delivery efforts. For example, if SaaS integration is the initial API challenge, your API taxonomy might focus first on application APIs. One Asian bank's API taxonomy has three major types: product/service APIs (which correlate to Forrester's notion of business APIs), interface APIs (business APIs exposed to partners, clients, and fintechs), and integration APIs. A global business and consumer services firm started with only a two-part taxonomy: business APIs and integration APIs. Intel has a more developed taxonomy with six major types: business process, master data, transactional data, UI component, security, and utility.[8]

In API strategy and governance workshops, Forrester uses a variety of representations to illustrate different aspects and structures of an API taxonomy. These serve as starting points for defining and communicating your API taxonomy:

In API strategy and governance workshops, Forrester uses a variety of representations to illustrate different aspects and structures of an API taxonomy.

› **Overall definition of API categories.** Four major categories correlate to four major usage scenarios (see Figure 2). Although the figure may depict internal APIs as a layer beneath open web and B2B APIs, they may in fact be the same APIs in all three cases, just with different policies applied.

› **Overall definition of API types.** To represent different purposes and scopes, your API taxonomy should include different API types (see Figure 3). Within each, it may be helpful to further specialize and refine Forrester's sample model. API types are a key foundation for tuning API governance processes. For example, business APIs should receive a high level of governance focus because, as the embodiment of an organization's digital business capabilities, they have broad impact.

› **Business domain models.** Although Forrester usually places business domains within the context of API portfolio management, not API taxonomy, it may be helpful to connect the two as you document and communicate your taxonomy. PayPal organized its APIs using a business capability model with seven or eight domains and more than 70 separate capabilities across the domains (see Figure 4).[9]

› **Relationships between API layers.** By placing different API types in a layered architecture diagram, you can help teams understand how each adds value to the overall portfolio (see Figure 5).

› **API taxonomy mall map.** As your API taxonomy evolves to have a number of API categories, types, and layers, it may be helpful to produce an overall map — like those used in shopping malls — to communicate and relate all of the taxonomy in one large diagram (see Figure 6).

FIGURE 2 Four Major Usage Scenarios; Four Major API Categories

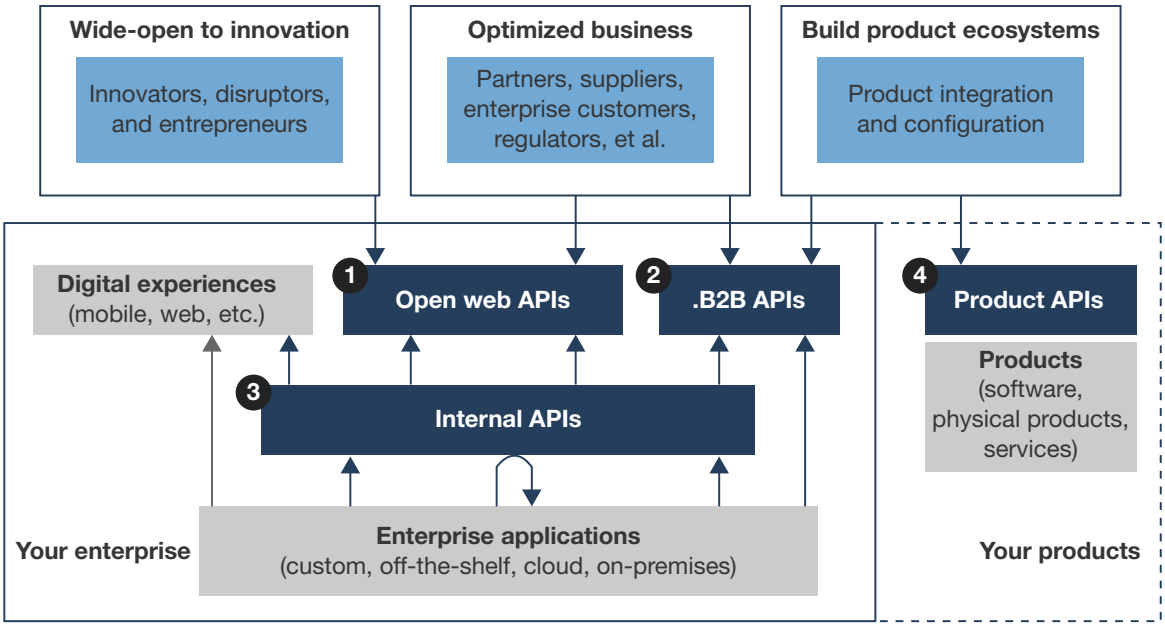


FIGURE 3 Purpose And Function Drive Distinctions Between Different Types Of APIs

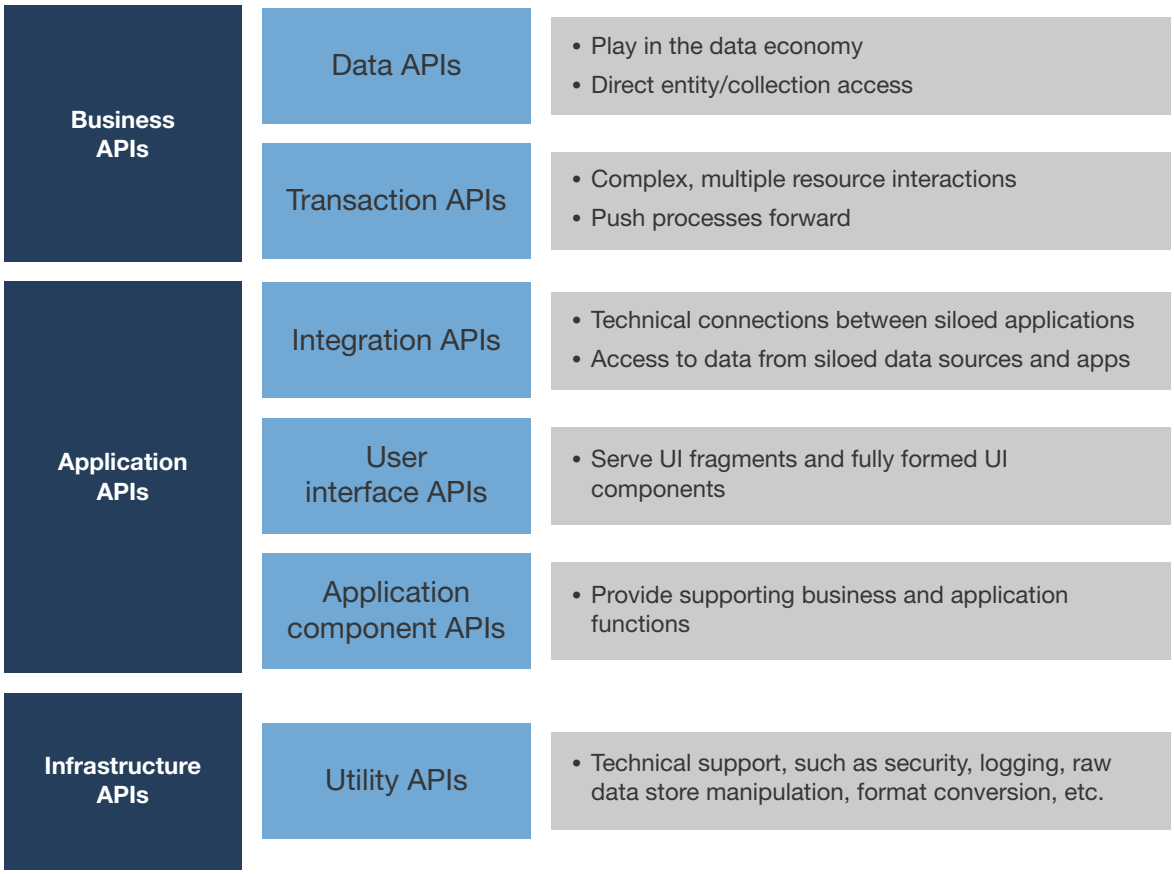
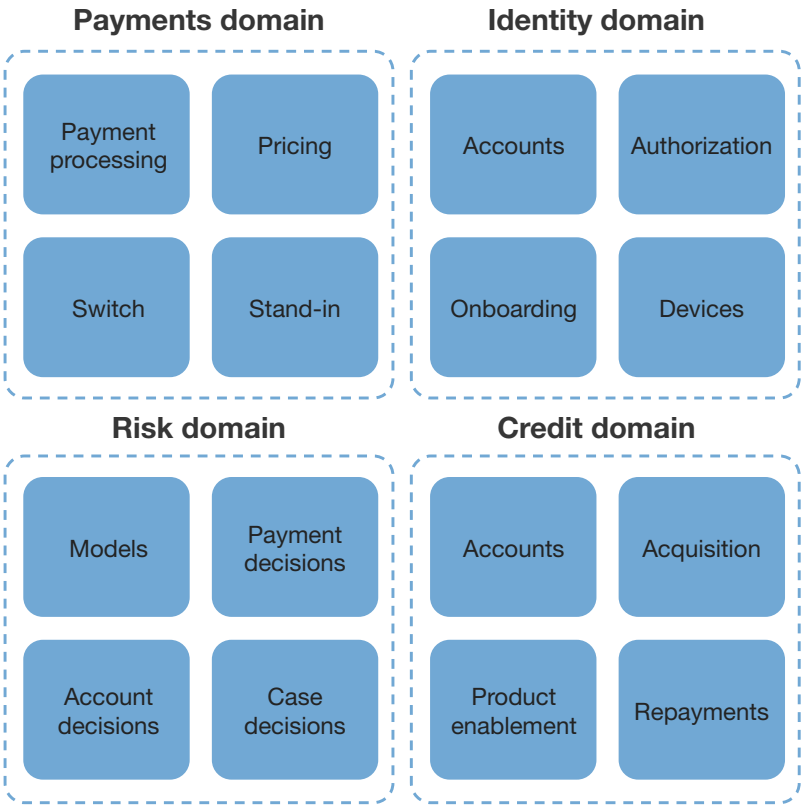


FIGURE 4 PayPal Uses Business Capability Modeling To Identify Its Business APIs, Organizing Them Into Domains



Source: InfoQ and PayPal

FIGURE 5 Layered Representation Of Your Taxonomy Helps Teams Understand Relationships Between API Types

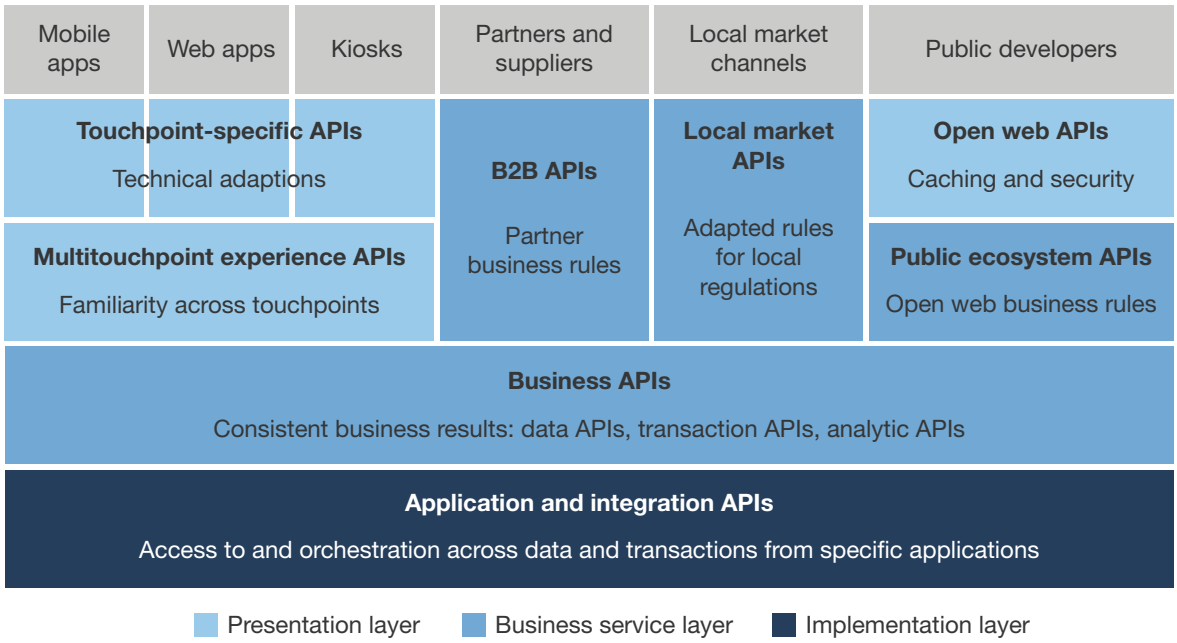
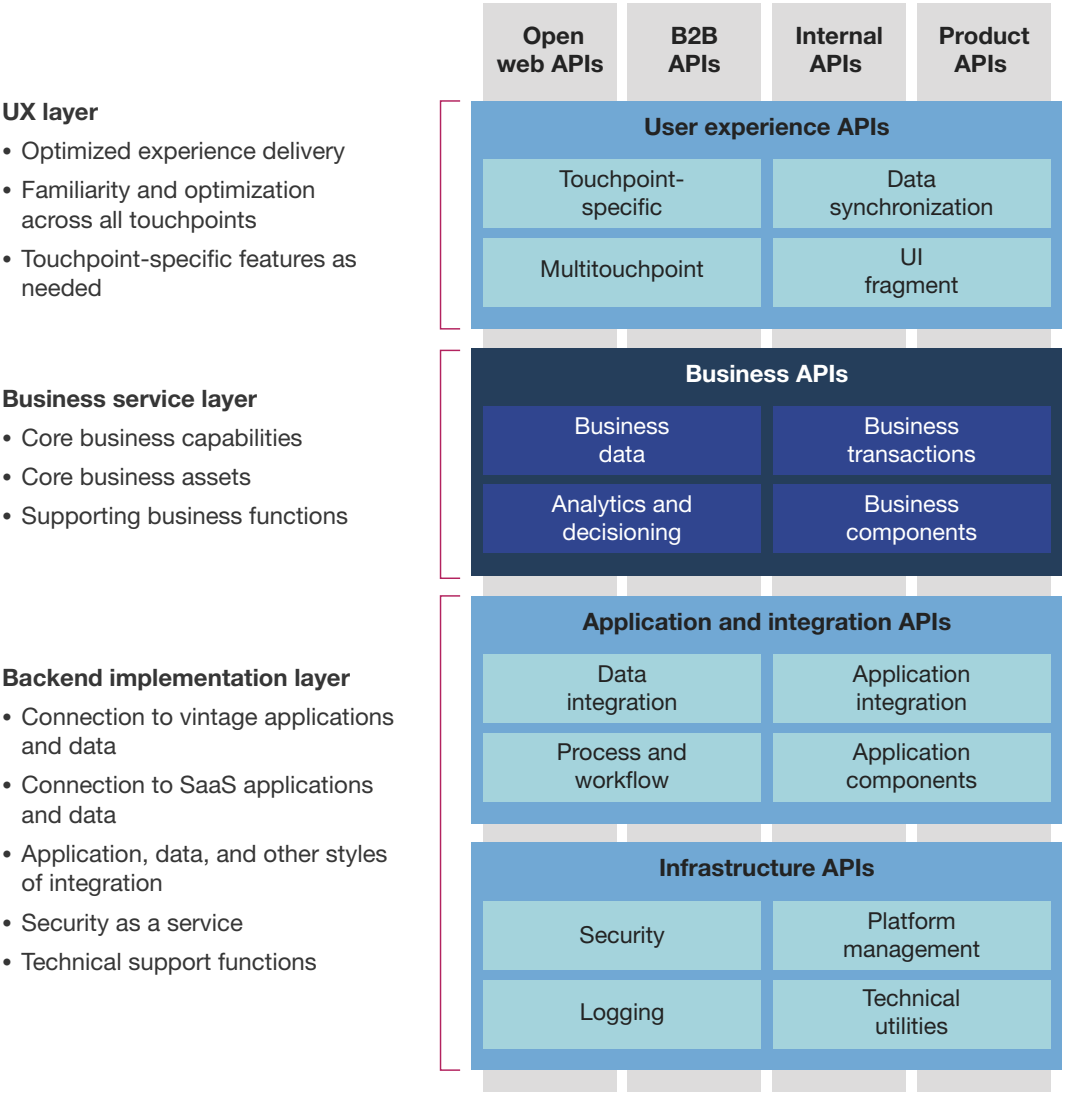


FIGURE 6 An Overall Map Of An API Taxonomy Can Help Teams Put All The Pieces Together



Recommendations

Agile-Plus-Architecture Aligns API Implementation With API Taxonomy

An API taxonomy helps to keep API identification, design, and implementation on track — *if* AD&D teams use it. If it's just an interesting picture on the wall, it's wasted effort. To prevent that from happening, use Agile-plus-architecture best practices to mix the right amount of architectural governance into Agile and DevOps processes.[10] Applying these practices to API taxonomy and API implementation, you:

- **Do lightweight API taxonomy work up front.** Rather than spending weeks trying to define a comprehensive API taxonomy at the start, define only the overall structure and major types.
- **Connect with project initiation efforts.** As you identify project work streams, whether through waterfall methods, Agile program initiation, or product planning efforts, help project teams develop their project-level API taxonomy (i.e., the specific subset of the organization's API taxonomy applicable to the project). Identify where and how to further refine the organization's taxonomy to meet project needs.

> Use Agile-plus-architecture best practices to mix the right amount of architectural governance into Agile and DevOps processes.

- **Connect with API design efforts.** As projects progress with APIs, collaborate to help design APIs according to the API taxonomy — or adjust the taxonomy based on project learnings.
- **Connect the API taxonomy to business architecture.** If your organization has a business architecture initiative, connect the business API elements of your API taxonomy to business capabilities you've identified in your business architecture and processes.
- **Connect the API taxonomy to API implementation patterns.** Catalog the ways your organization implements APIs (e.g., custom code, API gateway, enterprise service bus, composite orchestration, eventual consistency transaction management, etc.), then correlate which of these patterns are appropriate to use with different API types and scenarios.

Engage With An Analyst

Gain greater confidence in your decisions by working with Forrester thought leaders to apply our research to your specific business and technology initiatives.

Analyst Inquiry

To help you put research into practice, connect with an analyst to discuss your questions in a 30-minute phone session — or opt for a response via email.

Learn more.

Analyst Advisory

Translate research into action by working with an analyst on a specific engagement in the form of custom strategy sessions, workshops, or speeches.

Learn more.

Webinar

Join our online sessions on the latest research affecting your business. Each call includes analyst Q&A and slides and is available on-demand.

Learn more.

Forrester's research apps for iPhone® and iPad®
Stay ahead of your competition no matter where you are.

Endnotes

[1] Done right, APIs can open new angles into business strategy. They allow an enterprise to go beyond its traditional offerings to pursue new markets and customers by creating new products and services from its assets, data, or processes. APIs can also create new go-to-market strategies and new value for existing offerings, such as targeting customers through influencers rather than targeting customers directly. See the Forrester report "How APIs Reframe Business Strategy."

[2] As industry disruptions go, new government regulations come with long lead times, but executives still can choose shortsighted responses focused solely on regulatory demands. A better response is to use changing regulations — and other disruptions — as opportunities to advance API strategy and digital business transformation. See the Forrester report "APIs Turn Disruptions Into Business Opportunities."

[3] Forrester lays out a model of eight central API maturity areas — ranging from portfolio management and design strategies to development life cycles and funding. See the Forrester report "Drive Business Agility And Value By Increasing Your API And SOA Maturity."

[4] APIs have the power to unlock new revenue streams, transform how you design and deliver change, and extend your value proposition via dynamic ecosystems of value. But digital business design is a highly technical domain awash with jargon and misconceptions and one in which the underlying technology capabilities fundamentally drive

and constrain business model design. Neither eBusiness nor technology management teams can build a successful API strategy in isolation — collaboration is key. See the Forrester report "Brief: Four Ways APIs Are Changing Your Business."

[5] REST: representational state transfer.

[6] It's clear that mobile apps need APIs to access business data and transactions. The problem for AD&D pros is how to design these APIs. Building on Forrester's four-tier engagement platform model, we provide 14 key guidelines for the process, practices, and design principles that place mobile APIs into an omnichannel and enterprise context. See the Forrester report "How To Design APIs For Mobile."

[7] Successful delivery of APIs requires AD&D professionals to have the right strategy and the right infrastructure. API gateways and API management solutions are important but not sufficient. As a foundation for crafting a strong API platform strategy, Forrester defines and describes five major elements of a comprehensive API platform. See the Forrester report "Defining A Platform For API Success."

[8] Source: "An API Journey for Velocity and Lower Cost," Intel, April 5, 2016 (https://software.intel.com/en-us/articles/an-api-journey-for-velocity-and-lower-cost).

[9] Source: Erik Hogan, "Untangling an API-first Transformation at Scale. Lessons Learnt at PayPal – Part 2," InfoQ, February 23, 2017 (https://www.infoq.com/articles/paypal-api-first-part2).

[10] Agile development practices and continuous delivery are essential tools, but so is an architecture that enables resilience. Combining Agile and architecture is challenging. Agile development and continuous delivery help enterprises change quickly, but more than that, organizations need the ability to quickly and continually make one change after another after another. In other words, they need the benefits of both Agile delivery and architecture. See the Forrester report "Best Practices For Agile-Plus-Architecture."

FORRESTER®
CHALLENGE THINKING. LEAD CHANGE.

We work with business and technology leaders to develop customer-obsessed strategies that drive growth.

PRODUCTS AND SERVICES

- Core research and tools
- Data and analytics
- Peer collaboration
- Analyst engagement
- Consulting
- Events

Forrester's research and insights are tailored to your role and critical business initiatives.

ROLES WE SERVE

Marketing & Strategy Professionals

CMO
B2B Marketing
B2C Marketing
Customer Experience
Customer Insights
eBusiness & Channel Strategy

Technology Management Professionals

CIO
- Application Development & Delivery

Enterprise Architecture
Infrastructure & Operations
Security & Risk
Sourcing & Vendor Management

Technology Industry Professionals

Analyst Relations

CLIENT SUPPORT

For information on hard-copy or electronic reprints, please contact Client Support at +1 866-367-7378, +1 617-613-5730, or clientsupport@forrester.com. We offer quantity discounts and special pricing for academic and nonprofit institutions.

ORACLE®

Financial Services

The Oracle Cloud offers complete SaaS application suites for ERP, HCM and CX, plus best-in-class database Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) from data centers throughout the Americas, Europe and Asia. For more information about Oracle (NYSE:ORCL), please visit us at oracle.com.

Microsite: Build the Open Bank of the Future

Contact us: financialservices_ww@oracle.com

Connect With Us: