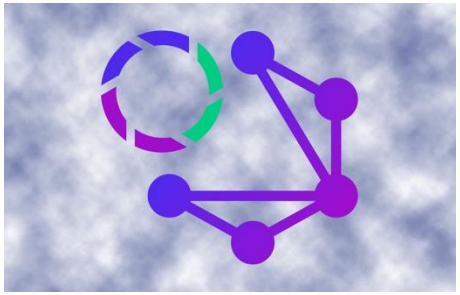
Modern approaches to headless and composable ecommerce

An Intellyx Analyst Guide for Nacelle

By Jason Bloomberg and Jason English

Table of Contents

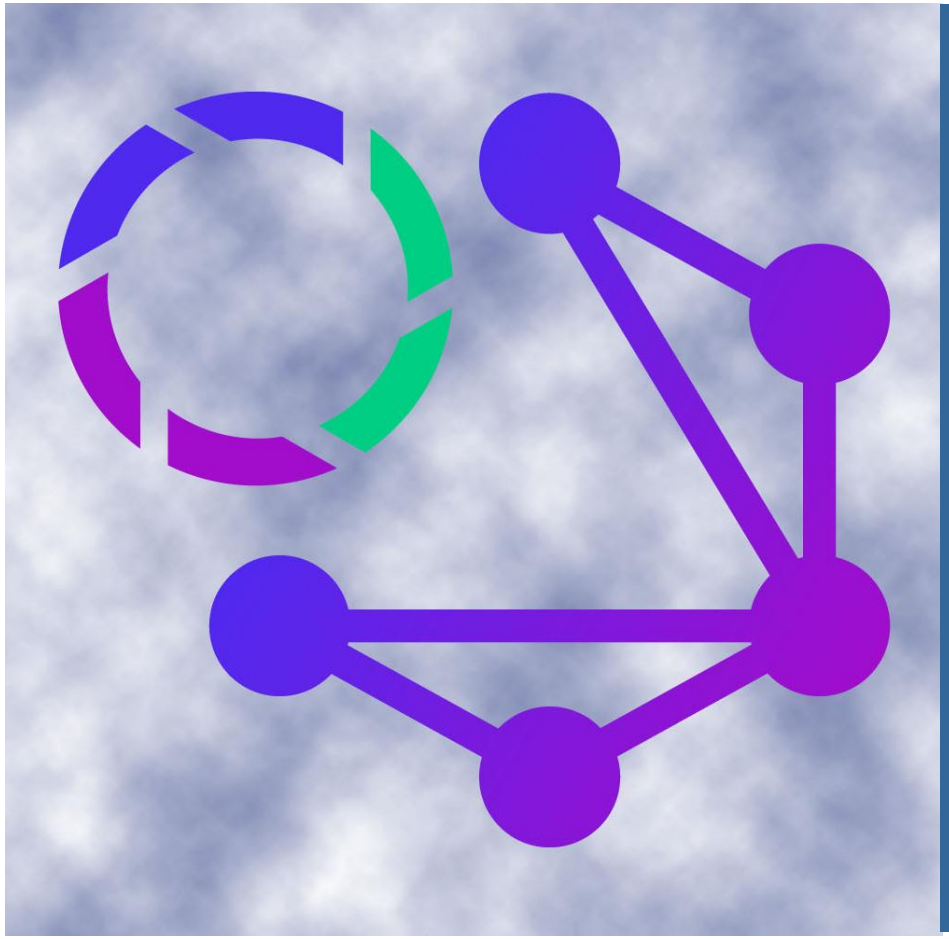



Introduction

Building a successful commerce application that delights customers takes much more than great imagery and glowing reviews. It requires a composable approach that truly separates front-end presentation layer concerns from back-end considerations of dynamic data and highly performant infrastructure.

A modern 'headless' approach should rapidly surface and deliver real-time product information, promotional offers, pricing, inventory, and order status directly to the customer's fingertips, in order to assure many happy returns.

This Intellyx Analyst guide will demonstrate how modern eCommerce websites and apps can benefit from the canonical data model of GraphQL, combined with the headless commerce capabilities of the Nacelle platform to relieve the heavy lifting usually required for building responsive commerce applications.

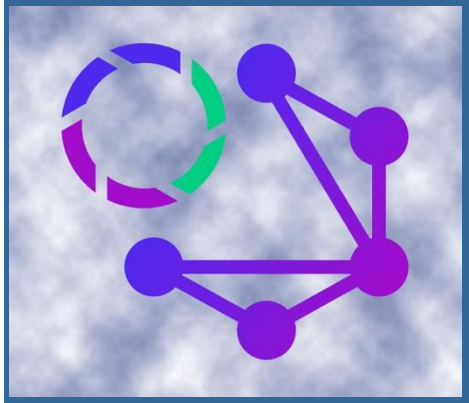


By Jason Bloomberg

Managing Partner & Analyst
Intellyx

The power and challenges of GraphQL

Part 1 of Modern approaches to headless and composable ecommerce



GraphQL is a query language for APIs and a runtime for fulfilling those queries with your existing data. GraphQL provides a complete and understandable description of the data in your API, gives clients the power to ask for exactly what they need and nothing more, makes it easier to evolve APIs over time, and enables powerful developer tools.

– GraphQL.org

Since Facebook released the GraphQL spec to the world in 2015, people's opinions about it have been dramatically divided.

Giving clients the power to ask for what they need from servers – perhaps the most important phrase in the standard definition above – represents a paradigm shift in how software implements integration in distributed environments.

On the other hand, GraphQL's detractors point out that implementations of the language fall short of its promise and that in real-world scenarios, performance and complexity limitations outweigh its strengths.

Both sides have a point. To get to the bottom of this controversy, we need to place GraphQL in context. How does it differ from integration paradigms that came before? What problems remain, and how can we solve them?

On the shoulders of giants

GraphQL owes much of its design to the two integration paradigms that preceded it: SOAP-based Web Services and REST. Each of these approaches solved a raft of problems while leaving others unresolved. Just as REST addressed many issues with Web Services, so too does GraphQL resolve problems with REST.

To understand the context of GraphQL, let's look at the table below.

	SOAP-based web services	REST	GraphQL
Paradigm	XML-centric	HTTP-centric	Request-centric
Endpoints	Discoverable endpoints	Resources at URLs	Single endpoint
Joins	Static; predefined in ESB	At client via multiple requests	Needs workaround
Performance bottleneck	XML parsing and transformation	Multiple round trips	Inefficient APIs

Table 1. Comparing GraphQL to web services and REST (source: Intellyx)

Paradigm

SOAP-based Web Services arose in the early 2000s, leveraging the text-based XML standard that became prominent in the 1990s. Abstracting arbitrary endpoints via XML provided a simple, common interface standard that was both technology and language-independent.

REST, in turn, leveraged HTTP to provide a lightweight, web-friendly approach to integration that resolved many of the challenges of Web Services. REST heralded a new era of simplified programmability for integration.

However, Web Services and REST require developers to code endpoint behavior on the server. Clients could only request data the server was already prepared to deliver. GraphQL turned this pattern around. With GraphQL, the request specified the data the client required – and it was up to the server to return just those data, no more and no less.

Endpoints

Instead of hard-coding clients to server endpoints, Web Services sought to make them discoverable. Clients could look them up in a registry and bind to them at runtime – at least in theory.

In practice, however, discoverability proved too difficult to be practical in most situations. REST solved this problem in a dramatic fashion by assigning any server endpoint a URL (or more generally, a URI).

Binding to an endpoint was as simple as performing a GET operation on a URL – something the web was already very good at.

Where REST fell short, however, was in situations where the client needed data from multiple URLs, requiring multiple round trips to the server.

Now it's GraphQL's opportunity to shine, as a GraphQL API requires only a single endpoint to handle any queries from clients.

Joins

Combining data from different sources has always been a knotty distributed computing problem. Web Services required developers to build joins on the ESB, creating SQL query strings to send to the underlying database.

REST added the ability for clients to perform joins at the cost of multiple queries to the server as well as processing and memory burdens on the browser.

GraphQL doesn't have a comprehensive answer to the problem of joins. Without the appropriate server-side infrastructure to support them, clients must select a workaround to combine data from different sources.

Performance Bottleneck

Every distributed computing deployment has some bottleneck – one part of the implementation that slows everything else down. With Web Services, it's the overhead of parsing and transforming XML. REST suffers under the burden of multiple round trips between client and server.
GraphQL, in turn, suffers from inefficient APIs. Typical implementations of GraphQL leverage REST under a façade, returning far more data from the back-end sources than the client requires. The client gets the data they want but at the cost of GraphQL server memory leaks, poor performance, and scalability issues that lead to errors.

Resolving the Challenges of GraphQL

Providing joins that follow the request-centric GraphQL paradigm, as well as the related problem of building efficient APIs, is now the crux of the GraphQL challenge.

These problems affect some GraphQL deployments more than others. Of particular concern: use cases that require joined data from multiple back-end data sources, for example, with e-commerce.

The solution is to build GraphQL infrastructure from the ground up to address the challenges of GraphQL – a challenge Nacelle addresses by optimizing how the back-end platform ingests, transforms, and indexes data for even the most complex GraphQL queries and joins.

The Intellyx Take

Both Web Services and REST dramatically simplified the challenge of integration while kicking the can down the road on specific problems that plagued each approach.

The proverbial can has now arrived at the feet of GraphQL. GraphQL's request-centricity promises to shift the power to the client where it belongs, especially for today's digitally savvy organizations with a renewed focus on customer needs.

The question for today: will the technology community kick the can down the road once more, leaving some future integration paradigm to address GraphQL's shortcomings?

Not if Nacelle has anything to do with it.

By Jason English

Partner & Principal Analyst
Intellyx

Solving the data assortment problem of digital merchandising

Part 2 of Modern approaches to headless and composable ecommerce

Before e-commerce, merchandising used to be the ultimate game-changer in retail.

Advertising and arranging an assortment of goods in a store, from the display window to the organization of product categories, to the planogrammed end caps and shelf spaces arranged from top-shelf to discount pricing — that's what used to set one retailer's brand and top-line revenue apart from the next.

Today's digital merchandising is in an entirely different league. It's like the difference between playing 2-D checkers and 4-D chess.

The customer customization conundrum

Today's consumers and business-to-business customers expect a digital experience that instantly offers products and services tailored to their needs, with an array of options and accessories delivered as quickly and cheaply as possible.

Confoundingly, this level of customization is hard to achieve because the sellers and suppliers needed to fulfill each transaction hardly resemble yesterday's retail value chain.

Virtually any product has a deeper digital footprint than its basic specifications and price. E-retailers can partner, resell, promote or make offers based on the metadata surrounding each product and its suppliers, as well as the metadata surrounding its potential buyer and their perceived preferences.

However, as an industry, we haven't gotten any smarter than our forebears about merchandising just because we've 'gone digital' or started talking about metadata. Retail chains have always maintained their audience metrics, sales histories and seasonal trends to help plan the assortment of SKUs and arrangement of each year's offerings. That hasn't changed.

Instead, the *rate of change* in customer preferences and demand is now happening much faster than any conventional merchandising plan can support.

Why is it so hard for today's digital merchandiser to overcome the physical world's limitations to reach customers with relevant options wherever they are?

Just-in-time manufacturing and marketing

The first wave of eCommerce in the late 1990s brought the supply chain concept of **just-in-time manufacturing** to the digital world. Companies like Dell thrived based on their ability to configure products to order with the help of a collaborative supplier network rather than maintaining huge inventory stocks or excess reserved capacity.

Fulfilling customer demand with build-to-order computers was a huge leap forward in efficiency and agility. Dell could market and feature products on their website that customers were likelier to buy, with order-to-promise dates that their manufacturers and logistics networks could deliver on.

But while this use case was certainly customer-centric, it was only valid for a vertically integrated high-tech value chain, where the enterprise buyer could dictate the specifications and data exchange standards at the center of the hub.

There are still modern merchandising lessons an enterprise could take away from this pre-Y2K just-in-time revolution, even if it's becoming clear that only the world's largest companies would be able to achieve leverage over their entire value chains, much less control the data within.

The data challenge of digital merchandising

A modern enterprise can focus on gathering more suppliers and vendor partnerships, hiring and retaining the best employees, or even a better website or storefront design – but none of these initiatives will make an impact if the data isn't ready to fulfill the customer's needs.

Have you ever noticed how a major convenience store chain like 7-11 generally arranges its assortment of products on the shelves in similar places, even if the store sizes and layouts are different?

They have retail merchandising down to a science – so much so that vendors supplying chips and soda receive purchase notifications and stockout alerts, then come in and fill their inventory from a truck stacked in each store's order and each shelf spot to be replenished.

If that sounds too complex to pull off, now try ***digital merchandising***—where the 'digital shelf space' for displaying combinations of products is theoretically limitless, but the customer's attention span is severely limited.

Data about a product now becomes the stand-in for a product in digital merchandising. *Some product data is easily standardized; for instance, a UPC barcode or number would scan as the same product in almost any warehouse or store shelf.*

In this world, many of the 'products' on display are not physical products at all – they may be different configurations of items, digital files, passwords, or tokens that give you access to a different price or a purely digital service such as a warranty or insurance. The data variations that can constitute a 'product' are infinite, as are the variations of how that 'product' can be displayed for different customers.

Applying a canonical data model for interoperability

With so much data variability, a commerce application should separate business logic and data from the presentation layer so that digital merchandising workflows can responsively present the right assortment of products and product options to customers.

Fortunately, many e-commerce companies and software vendors are improving as they publish their own APIs and addressable services for partners, preferably by applying a ***canonical data model***.

A canonical data model can break down all of the elements of a given product's data and metadata from its sources into discretely defined attributes that can be discovered and searched across multiple dimensions rather than providing a strict categorization model or hierarchy that could prove inflexible to advanced queries or recommendation engines.

That's great from a flexibility perspective, but there's still a variability problem to address, as sophisticated customers will continue to demand new features, and ask for more details in every purchasing decision.

The different catalogs, business process, automation and transactional vendors that feed an e-commerce app may have differing views on how to split the canonical data atom into its component parts.

The **Nacelle Composable Commerce** platform takes a novel approach to this problem. Its single GraphQL API sits on top of a highly performant database system that processes and normalizes. Despite being a single GraphQL endpoint, merchants can request data across all commerce and content domains in one query.

The Intellyx Take

Instead of arranging products in store windows and shelves hoping for foot traffic, digital merchandising arranges data in real time to meet the needs and preferences of customers, wherever they are.

The good news? A new value chain for commercial data has been in the works this whole time. Partners are breaking down their old data silos, from content suppliers enriching product catalogs with attribute data to transactional vendors confirming complex orders.

Companies are deploying headless e-commerce sites that separate the visual concept of a customer storefront from a highly responsive, canonical data layer that serves up exactly what the customer needs.

By Jason Bloomberg

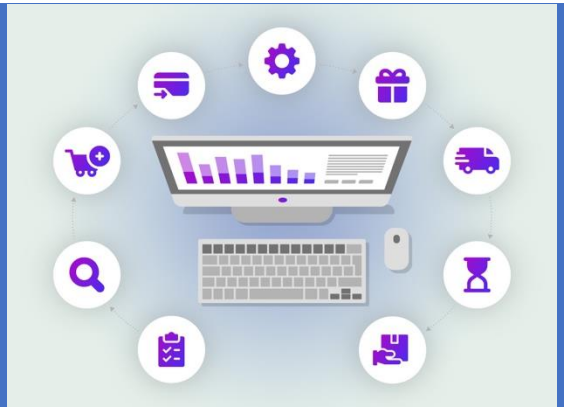
Managing Partner & Analyst
Intellyx

How to turn eCommerce infrastructure inside out

Part 3 of Modern approaches to headless and composable ecommerce

In part one of this four-part series, I laid out the strengths and weaknesses of the GraphQL API query language for fulfilling queries. Then in part two, my colleague Jason English discussed how vital a canonical data model is for modern e-commerce offerings.

English pointed out that Nacelle's Composable Commerce platform's request-centric GraphQL-based data engine [transforms unstructured data into structured data] processes and normalizes commerce and content data into a data layer that can respond to complex multi-dimensional requests with one API; which makes it much faster and more efficient than managing multiple APIs for each service.



It might appear that GraphQL is a magic elixir, able to fix all that's wrong with traditional monolithic e-commerce systems. GraphQL is powerful, to be sure – but there's no magic here. Let's delve deeper into Nacelle's platform to gain a better understanding of how it works.

Data ingestion and normalization

The Nacelle platform can ingest data from multiple sources (since it's system agnostic) – including best-of-breed systems for content, category, product, and inventory that are already in place in the e-commerce merchant's infrastructure.

Nacelle uses pre-built connectors to access the APIs of these systems, while also ingesting data from data lakes and even flat files when necessary.

The result of all this ingestion – much of it in real-time – is a mishmash of different data formats. This diversity of data is why normalization is so essential.

Nacelle's proprietary data orchestration layer handles this normalization by transforming incoming data into a single, consistent format. It then indexes that data to support rapid queries. Finally, it stores the data, making it available when and where the end-user wants it.

One of the primary benefits of normalization that differentiates Nacelle from other e-commerce solutions is the way it ensures that future changes to the data stack won't require a complete refactor of existing connected systems.

In other words, it's relatively simple to plug in additional data sources to enrich the e-commerce experience for end-users.

Data delivery for a flexible headless experience

Nacelle delivers both content and product data via a single GraphQL API. This API is where the power of GraphQL pays off for Nacelle customers, as the Nacelle platform automatically provides the information that any query is looking for – fast.

This data delivery supports multiple storefronts, apps and channels for a comprehensive e-commerce experience.

Supplemental to its GraphQL, Nacelle leverages software development kits (SDKs) and accelerators to support whatever off-the-shelf or customized front-end technology an e-commerce merchant wants to use.

Despite this diversity of modes of consumption, Nacelle's normalization logic supports all of them. There's no need to rewrite the normalization logic for each head or channel. Instead, Nacelle provides a consistent data structure across the board – for products, variants, content, inventory, and orders.

Furthermore, Nacelle propagates changes to all channels in real time. For example, if marketing launches a new campaign, Nacelle will transmit the specifics of that campaign to each head in a coordinated fashion.

Composability on the Front End

Nacelle's approach to composing back-end applications is data orchestration-based, rather than the typical integration-based approaches of earlier generation eCommerce platforms.

The data normalization creates the canonical data model for heads, endpoints, and channels to compose into working eCommerce solutions.

This data-centric approach turns traditional approaches to eCommerce infrastructure inside out. With Nacelle, merchants can compose any applications or data sources to create the eCommerce stack they desire – delivering the content and product data to the right customer at the right time.

Furthermore, Nacelle works with all the legacy commerce solutions on the market, including Magento, Shopify, Salesforce Commerce Cloud, and many others. Merchants don't have to migrate off an existing platform to take advantage of the flexible, customer-focused capabilities that Nacelle delivers.

This data-centric composability empowers merchants to optimize the front-end shopping experience, both in terms of the product data and content, but also with respect to the performance and efficiency of the eCommerce workflows – critical for reducing shopping cart abandonment.

Given the high expectations among customers for seamless, high-performance eCommerce in today's Amazoned world, Nacelle is well-positioned to help many different types of merchants become more competitive and profitable.

The Intellyx Take

Historically, headless eCommerce has largely overpromised and underdelivered. Delivering back-end eCommerce functionality via APIs to whatever front-ends a merchant might wish to implement makes perfect sense – but the devil is always in the details.

GraphQL is an important tool for addressing the limitations of headless eCommerce. Instead of the back-end delivering their data via multiple APIs, the front-end can request just the data it wants from a single API.

GraphQL alone, however, falls short without the infrastructure to support it. By ingesting data from multiple sources, normalizing it to create a canonical data model, and then providing the GraphQL API for diverse front-end technologies to query, Nacelle provide the infrastructure necessary to make the promise of GraphQL a reality for eCommerce merchants of all sizes.

By Jason English

Partner & Principal Analyst
Intellyx

Measuring the cost vs. value ratio of headless commerce

Part 4 of Modern approaches to headless and composable ecommerce

Retailers evaluate their performance in terms of total sales, new customer acquisition, and revenue per customer, offset by any sales costs, including the cost of the goods themselves, goods, and marketing and promotional expenses.

In practice, no matter how well a company does, there's always room for improvement. That's why such self-evaluations often result in the same mission: *"We need to build a better website to improve the customer experience."*

But what is making 'a better website' really going to take? Is it worth the cost and effort? And which features should we build first?

For a modern retailer, the Total Cost of Ownership (TCO) of a new e-commerce presence means the total labor, software, service and infrastructure cost of building and supporting a truly *responsive, dynamic, accurate data-driven website* that can quickly give customers the products and experiences they are looking for.

Reducing pipeline time for customer-facing features

In this series, we've discussed how GraphQL's unique ability to allow users to 'ask for what they need' from servers and services simplifies interoperability. We have explained how a canonical data model for assortments of products and offers can speed up digital merchandising and how Nacelle's approach to managing headless commerce with GraphQL speeds up integration.

Now we've come full circle as we seek to justify the value of our investments and prioritize developing future features that could influence customer experience.

Industrial science has long understood the "***bullwhip effect***," in which a business continually reacts and makes changes to policy due to current conditions like stock outs or abandoned orders. This thrashing back and forth throws the entire business out of sync as a wrong decision driven by a customer touchpoint cascades to affect upstream supplies, production and logistics capacity.

In e-retail terms, this effect is more than quickly supplying product information and customizing offers at the point of sale. It's about

shortening the pipeline or the length of the 'bullwhip' that can impact the timeliness of data and releases delivered to the customer now and in the future.

No retail website or web app is an island unto itself. Each customer session may pass through third-party API services and components for predictive recommendations, inventory, referrals, reviews, customization, and feedback – not to mention the site's performance and error-handling telemetry.

Retailers want the added customer experience offered by many marketing services, algorithms and partner plugins. However, each additional segment added to the pipeline takes its own bite out of customer response time and potentially creates a performance bottleneck that must be addressed.

So the time it takes to build a new site, the time it takes to make changes to the workflow in the future, and the time spent wrangling with data and components all contribute something to the cost of the site in terms of revenue and value gained or lost.

Add business agility

Nacelle's GraphQL architecture offers a flexible approach for making API requests across multiple backend systems. Using GraphQL, the underlying e-commerce system can change to meet customer needs without requiring rebuilding the front end of the storefront, or extending the delivery timeline to 'rip and replace' parts of the infrastructure.

This gives merchants a competitive edge by reducing technical debt, lowering switching costs, and increasing agility.

How do modern retailers realize value by implementing headless commerce?

Conversions and order size. You can't change the fundamentals of retail – it's still all about selling something. The primary value of making the exact right personalized offer in near real-time helped Enso Rings **achieve a 26.5% increase in order conversions and a 12.5% increase in value per order.**

Faster implementation time. Gimme Beauty went from concept to a total redesign and re-architecture of their commerce application in **just 90 days.** Even at that breakneck speed, the resulting improvements **increased their conversion rate by 20%, with zero downtime.**

API-driven development flexibility. Global florist network FTD took a platform-first approach to modernizing its built-in-house website to improve developer efficiency. Using Nacelle's headless commerce approach, developers compose progressive web applications that make API calls to fetch product and commerce data, as well as rich content and media.

Site speed and conversion rates were improved, but more importantly, the company addressed technical debt with a new system that allows quicker UI/UX experimentation, earlier testing, and faster releases to meet customers with improved functionality.

The Intellyx Take

When considering the standard metrics like TCO and ROI for modernizing a web property, it's essential to take a step back and consider the potential costs and risks of not moving to a truly dynamic headless or composable e-commerce system.

To compete, retailers may continually add new features, specials and offers in an attempt to retain customers. If those new features create interminable wait times, customers will search elsewhere for answers.

Fortunately, we are experiencing a renaissance of computing power, bandwidth, automation, and the advent of very flexible, high-performance API-driven data layers like GraphQL. Forward-thinking retailers can now flexibly deliver on the promise of headless commerce to improve customer experience without the pain or sticker shock of a big-bang rip-and-replace implementation.

About Jason Bloomberg

Jason Bloomberg is founder and managing partner of enterprise IT industry analysis firm Intellyx. He is a leading IT industry analyst, author, keynote speaker, and globally recognized expert on multiple disruptive trends in enterprise technology and digital transformation.

He is #13 on the Top 50 Global Thought Leaders on Cloud Computing 2023 and #10 on the Top 50 Global Thought Leaders on Mobility 2023, both by Thinkers 360. He is a leading social amplifier in Onalytica's Who's Who in Cloud? for 2022 and a Top 50 Agile Leaders of 2022 by leadersHum.

Mr. Bloomberg is the author or coauthor of five books, including Low-Code for Dummies, published in October 2019.

About Jason 'JE' English

Jason "JE" English is Partner & Principal Analyst at Intellyx. Drawing on expertise in designing, marketing and selling enterprise software and services, he is focused on covering how agile collaboration between customers, partners and employees accelerates innovation.

A writer and community builder with more than 25 years of experience in software dev/test, cloud computing, security, game development and supply chain companies, JE led marketing efforts for the development, testing and virtualization software company ITKO from its bootstrap startup days, through a successful acquisition by CA in 2011. He co-authored the book *Service Virtualization: Reality is Overrated* to capture the then-novel practice of test environment simulation for Agile development. Follow him on Twitter at @bluefug.

About Intellyx

Intellyx is the first and only industry analysis, advisory, and training firm focused on customer-driven, technology-empowered digital transformation for the enterprise. Covering every angle of enterprise IT from mainframes to cloud, process automation to artificial intelligence, our broad focus across technologies allows business executives and IT professionals to connect the dots on disruptive trends. Read and learn more at https://intellyx.com or follow them on Twitter at @intellyx.

About Nacelle

Nacelle is a composable commerce platform provider that allows brands and retailers to publish commerce and content data to multiple storefronts, apps and channels by transforming, storing and reindexing data in real-time. With Nacelle, companies can future-proof their business by composing the commerce stack they want — giving them the agility needed to build unique and dynamic shopping experiences while optimizing business operations for growth.

For more information, go to nacelle.com.