entando

IT LEADERS ALERT

Choosing the Right Technology Platform is Crucial for the Success of Your Composable Application Strategy

By Massimo Pezzini, Independent IT Advisor

Table of Contents

CHAPTER 01 Introduction

CHAPTER 02 What Are Composable Applications and Why Are They Important?

2.1 Composable Applications Consist of Agile Orchestrations of Reusable Assets
2.2 Composable Applications Provide Benefits to Both Business and IT

CHAPTER 03 What Should You Do to Prepare for Composable Applications?

CHAPTER 04 Who is Involved in Application Composition?

4.1 Contributors to a Composable Application
4.2 An Emerging Approach to Composition: The Fusion Team

CHAPTER 05 The Application Composition Platform

5.1 Application Composition Platform Benefits
5.2 How Do You Choose an Application Composition Platform?

CHAPTER 06 How to Get Started with Composable Applications

CHAPTER 07 Conclusions

Massimo Pezzini is an independent IT strategic advisor. Mr. Pezzini's focus is on composable and digital business enabling architectures and technologies, including application and data integration; hybrid integration platforms (HIP); integration PaaS (iPaaS); APIs; events, digital integration hub (DIH); application composition technologies; and in-memory computing (IMC).

Mr. Pezzini has decades of experience in distributed computing, middleware technology and software architectures. Previously he was VP Distinguished Analyst and Research Fellow Emeritus with Gartner, inc. In that role he published hundreds of research documents, presented in dozens of Gartner's and third-party conferences and provided strategic technology advice to hundreds of large and midsized user and vendor organizations worldwide.

Massimo Pezzini,
Independent IT Advisor

CHAPTER

01

Introduction

Business agility and rapid response to opportunities and threats are imperative for organizations if they want to survive in the current, highly unpredictable, fast-changing business environment. As an IT leader, you can fulfill these imperatives by applying the composable application approach for IT and business teams to collaborate, build, maintain, and continuously adapt business-critical applications in a highly responsive way.

Adopting an Application Composition Platform (ACP) will provide you with the business agility and technology foundation needed to take advantage of the trend toward composable applications quickly and successfully. The seven most important criteria for choosing an ACP are discussed in section 5.2 of this document.

CHAPTER

02

What Are Composable Applications and Why Are They Important?

For an IT leader like you, the only certainty is uncertainty. Your organization's business priorities may, and most likely will, change suddenly and unexpectedly. You may have to abruptly downsize, postpone, or cancel strategic initiatives you are responsible for and shift your attention to previously lower-priority projects. The goals of some of these initiatives may change rapidly or you may have to realign resources you hadn't planned for. And of course, your budget will remain the same, grow modestly, or be cut back.

Therefore, your imperative is to adopt methods, approaches, architectures, and technologies that support business agility by quickly delivering new applications and adjusting their functionality to respond to these changing business needs, whilst keeping costs and the size of your teams under control.

Agile methods; technologies such as as cloud computing, low-code and pro-code platforms, DevOps, CI/CD, APIs, events, and containers; and architectures like micro frontends, microservices and multichannel come to mind as enablers for responsive business agility. However, an overarching approach combining and extending these concepts, making them accessible and affordable by mainstream organizations is gaining popularity and early adoption. This approach is referred to as composable applications.

The overall goal of composable applications is to enable your teams to dramatically reduce time-to-value for new applications, enable business people to be an active part of the development process, and streamline the applications' ongoing lifecycle management. The core principle of this approach is that applications are not built from scratch, but assembled by leveraging pre-built reusable assets.

You can think of composable applications as an evolution of the well-known API economy paradigm. The API economy extracts business value from a relatively small number of the organization's IT capabilities (functionality and data) by encapsulating them into reusable services accessible via APIs. This way, business teams and partners can leverage (and reuse) these capabilities by incorporating them in their applications and processes as added value components.

The composable approach pushes the API economy paradigm one step further by assuming that if a growing portion of your application portfolio consists of modular, reusable and easily composable capabilities, then your organization's ability to generate genuine new business value in a short time-to-value manner is multiplied manifold. Basically, a composable application strategy would enable your organization to transition from the classic coarse-grained API economy to a much more fine-grained "capability economy".

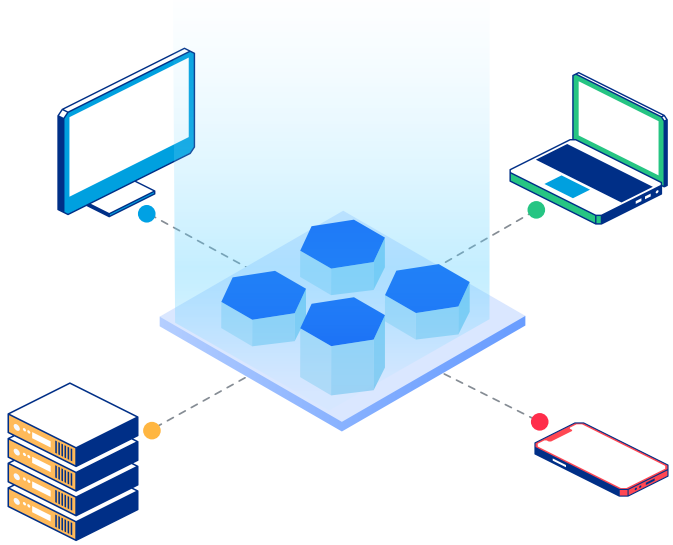
2.1 Composable Applications Consist of Agile Orchestrations of Reusable Assets

Composable applications are defined as orchestrated assemblies of independently deployable building blocks, supporting multiple custom user experiences. Composable applications can also be rapidly changed and extended via re-composition (see figure 1).

According to the composable approach, your organization's application portfolio would consist of an array of modular, granular, and reusable functional building blocks (sometimes referred to as packaged business capabilities [PBCs]), each implementing a well-defined, self-contained, and autonomous business function, and exposing well-documented APIs and/or event channels.

Besides these functional building blocks, the portfolio of reusable assets includes also frontend building blocks such as data entry forms, design system templates, dashboards, bots and miniapps. All these building blocks can be used and reused by your development, or rather "composition" teams, alongside alongside the functional building blocks to effortlessly, compose new applications.

The building blocks are fully documented in a searchable catalog (also called a marketplace or hub), so that developers can find those relevant to a specific task and "compose" them to implement yet another composable application. Developers can decide that even the composable applications themselves and other elements (e.g., business rules, data schema, orchestration models, and configuration files) can be documented and added to the catalog, so that they also become reusable assets that can be leveraged for the composition of other applications, even by other teams.



- Orchestrated Assembly
- Indipendently deployable Business Building Blocks
- Suppoting Multiple, custom UX's
- Extensible, Changeable via recomposition

Figure 1 – Composable Application Defined

The composition teams compose new applications, with end-to-end modularity, by assembling the appropriate combination of reusable assets, typically using low-code orchestration tools.

Composable Applications vs. Microservices

Are composable applications synonymous with microservices? The answer is no. However, the microservices principles fit very nicely with the composable architecture tenets: modularity, discoverability, independence, and reusability. In most cases, your teams will develop functional building blocks as combinations of API-enabled, container-based microservices.

Nonetheless, your teams could also develop functional building blocks as API-enabled pieces of traditional monolithic application software, not leveraging container technology, e.g., as COBOL/CICS transactions running on an IBM zOS mainframe or a set of RPG programs running on an IBM midrange system. Such an approach would not align with mainstream industry trends, but wouldn't violate any composable application principle.

Similarly, micro frontends (MFEs), the frontend counterpart to microservices, fit quite naturally with the composable architecture tenets. Your teams can develop frontend building blocks as collections of MFEs. However, this is not mandatory according to composable application principles. To build frontend building blocks, your teams could legitimately use any other approach, like Jamstack or declarative UI frameworks, if it fits with your needs better than MFEs, as long as it follows the tenets of the composable approach in terms of modularity and reusability.

Asset Reuse

In the context of a given composable application, a composition team can reuse an asset in multiple ways:

As is

The team assembles the asset with others without any modification, customization or configuration.

Configured

The team configures the asset according to their own requirements via the asset configuration tool. For example, a building block may allow developers to configure whether dates are in the MM/DD/YYYY or DD/MM/YYYY format by means of a configuration file.

Forked

The team modifies the asset, creating a new one, which they may document in the catalog so that other teams can reuse it.

Extended

This is a variant of the forked reuse case where the team removes limitations or bugs in the asset and creates an upgraded version. In this way, all the composite applications using this asset automatically inherit these improvements, taking into account the implied change and version management issues.

2.2 Composable Applications Provide Benefits to Both Business and IT

The benefits of composable applications stem from their focus on modularity, reuse and collaborative development. Compared with conventional monolithic systems, composable applications provide the following categories of benefits.

Support for business agility and faster time-to-value:

This is the primary reason why organizations look at the composable application approach. Its emphasis on reuse, coupled with the use of low-code application platforms, leads to faster and more flexible development. This means that you can significantly improve business agility, but also accelerate time-to-value for new innovations.

Additionally, composable applications support business agility and rapid response because they are easier to change and maintain than classic monolithic applications. For example, your team could change the application by simply modifying the model describing the relevant orchestration, adding a new approval step, or changing a validation rule without disruption.

Greater consistency in user experience:

If developers implement a certain functional or frontend building block as a reusable asset, all the composable applications leveraging that asset will deliver a consistent experience.

Faster and scalable innovation cycle:

If you have multiple composition teams in your organization, each can quickly and independently develop and implement composable applications that enable innovative products, services, or business processes. The result is faster and more widespread innovation cycles as opposed to the classic centralized approach, which is limited by what a single team can deliver in a given amount of time.

Facilitated multi-sourcing:

Composable applications favor a multisource approach to accrue reusable assets. You could buy them, build them in-house, commission their development to external service providers, or use any combination of these methods to quickly amass your own library of “commodity” and “differentiating” assets.

Lower implementation and maintenance costs:

Functional and frontend building blocks are, by definition, reusable across multiple composable applications. In principle, every asset used in composable architectures is reusable, including the applications themselves, which helps teams optimize initial implementation costs. Moreover, modularity and the related implicit separation of concerns help accelerate testing, problem isolation, and debugging. Finally, when a composition team fixes a bug or extends the functionality of a reusable asset, all the composable applications that utilize that asset automatically inherit the improvements, thus helping to reduce maintenance effort, time, and cost.

Greater business and IT alignment:

Your business and IT teams can work together collaboratively, to develop or enhance the composable applications. This way, misunderstandings and lost-in-translation issues between business and IT are minimized and the feedback loop is much more efficient and effective.

Separation of concerns:

The composable application approach, with a strong focus on modularity, allows you to optimize the development teams' work allocation, as well as application testing and debugging. For example, you could allocate the development of functional building blocks to hardcore business logic expert developers, the frontend blocks to UX designers, and the orchestration of these components to business process experts. This way, you optimize scarce development resources, keep costs under control, and further reduce time to value by having multiple specialized teams working in parallel.

Cleaner application architecture:

The principles of modularity, autonomy, loose coupling, and late binding (via APIs and events) that underpin a composable application embody some of the software architecture industry's best practices, aligning composable applications to the most proven, popular, and widely supported application design patterns.

Asset Sourcing

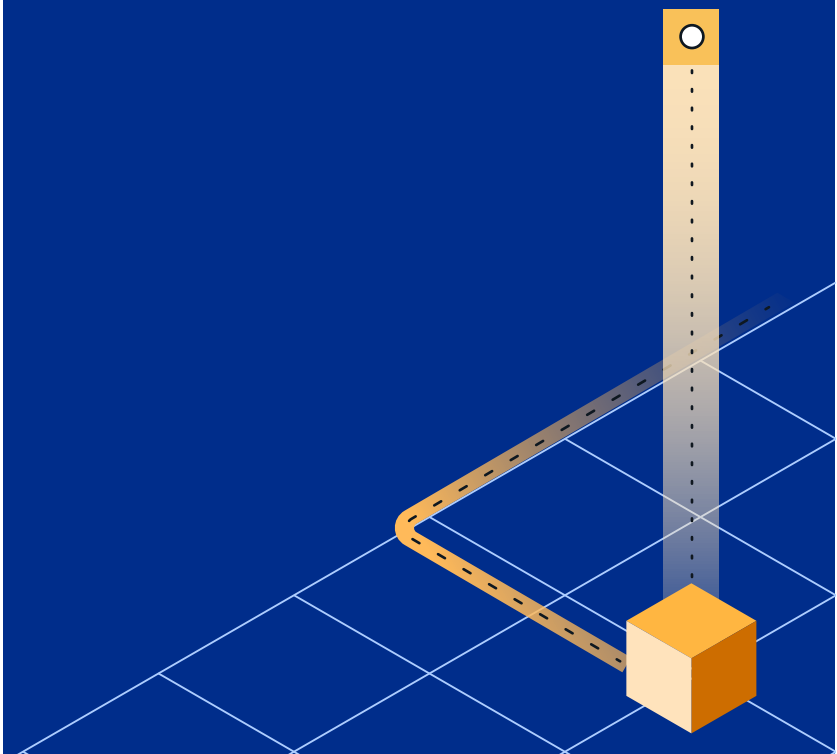
There are multiple ways in which your composition teams can source reusable assets:

Build new assets from scratch, with multiple teams working in parallel. For instance, one team develops functional building blocks while another one constructs the frontend building blocks.

Obtain functional building blocks by wrapping some of the functionality of your organization's pre-existing monolithic applications into APIs and/or event channels.

Outsource asset development to external service providers.

Purchase collections of functional building blocks from ISVs, typically in the form of "headless" SaaS applications.



CHAPTER

03

What Should You Do to Prepare for Composable Applications?

Although it is perfectly possible to apply the composable approach in the context of a single specific initiative, the benefits discussed above can be fully realized when composability is adopted strategically across multiple ventures. Gradually and consistently this approach should become the default for implementing new business systems, thus spreading its benefits and sustaining the trend toward the democratization of IT.

To begin adopting methods for composable applications, as an IT leader you should take the following actions:

Validate the architecture, approach and enabling technologies. The composable application architecture and approach, and some of the associated technologies may be new for many organizations. Likely, your organization will have to go through a learning curve to get up to speed. Therefore, before embarking on the journey, you may want to test the waters with a POC or POV to identify organizational and technology investments that need to be aligned. The POC/POV will also help to articulate the benefits of the approach from both the business and IT perspectives.

Educate business leaders about the potential benefits of composable applications. Alert them to the need for business leaders and their teams to be much more involved in the process, not only in the requirements gathering phase, but also in the actual implementation and ongoing maintenance of the applications designed to meet their needs.

Incrementally align business and IT teams by developing a common language through the concept of building blocks. Developers should design and document these assets so that your business teams can easily understand the functional capability of each (for example, tax calculation, shopping cart management, customer onboarding). In this way, your composition teams can decipher the module's role in the business process and how it relates to other building blocks. Ideally, business users should be able to understand what a building block provides functionally in the same way they can intuitively figure out which menu item they have to click for access to a certain business function (for example, “enter new purchase order”).

Establish appropriate catalog-based governance processes, which require the implementation and management of a shared catalog where reusable assets are properly documented. Then your composition teams can turn to the catalog to search, evaluate and reuse assets. The catalog is also used to enforce asset standardization, track utilization, and manage their lifecycle. The catalog becomes the locus where governance processes intersect with your teams’ development, QA, security and compliance work. In this context, it is critical to strengthen your team’s functional, QA and integration skills. This is needed to effectively support a multi-sourcing approach to building blocks.

Deliver a comprehensive composable technology foundation, which provides the capabilities that different composition team members or contributors (see section 4.1) need to fulfill their role. These capabilities are accessible through a UX that is in line with their skills, experience and expectations. Your IT department is responsible for designing, implementing, supporting, maintaining and advancing such a foundation. It should be delivered in a shared environment which can be accessed by all the contributors in a self-service fashion.

CHAPTER

04

Who is Involved in Application Composition?

Developing composable applications requires collaborative effort from different players, all of whom can be referred to as “contributors”. They usually work in composition teams, each in charge of delivering applications targeting specific business needs.

4.1 Contributors to a Composable Application

Contributors in a composable application development process typically include:

Enterprise architects with technology architecture skills, who are in charge of validating and advocating for the composable application architecture, selecting the appropriate set of enabling technologies, and validating them through POCs or POVs.

Creators such as solution architects, application architects, and professional developers, who design, implement and maintain the reusable building blocks.

Composers like solution architects, application architects, professional developers, and business technologists (business people with IT skills), who build new and upgrade existing composable applications by combining and leveraging reusable assets.

Enterprise architects with business process skills, who produce business capability maps and value streams that provide the conceptual blueprint for organizing the building blocks portfolio and aggregating them into domains.

Curators who essentially manage and govern the composable application environment. First and foremost, they administer the asset catalog, organizing and managing the assets stored there, effectively turning them into reusable components. The curator’s role also includes responsibility for the operations of the technology foundation. In many cases, responsibilities for “catalog management and governance” and “technology operations” are split between members of the curator community with different skill sets.

Consumers, the business users who utilize the composable application to perform their job.

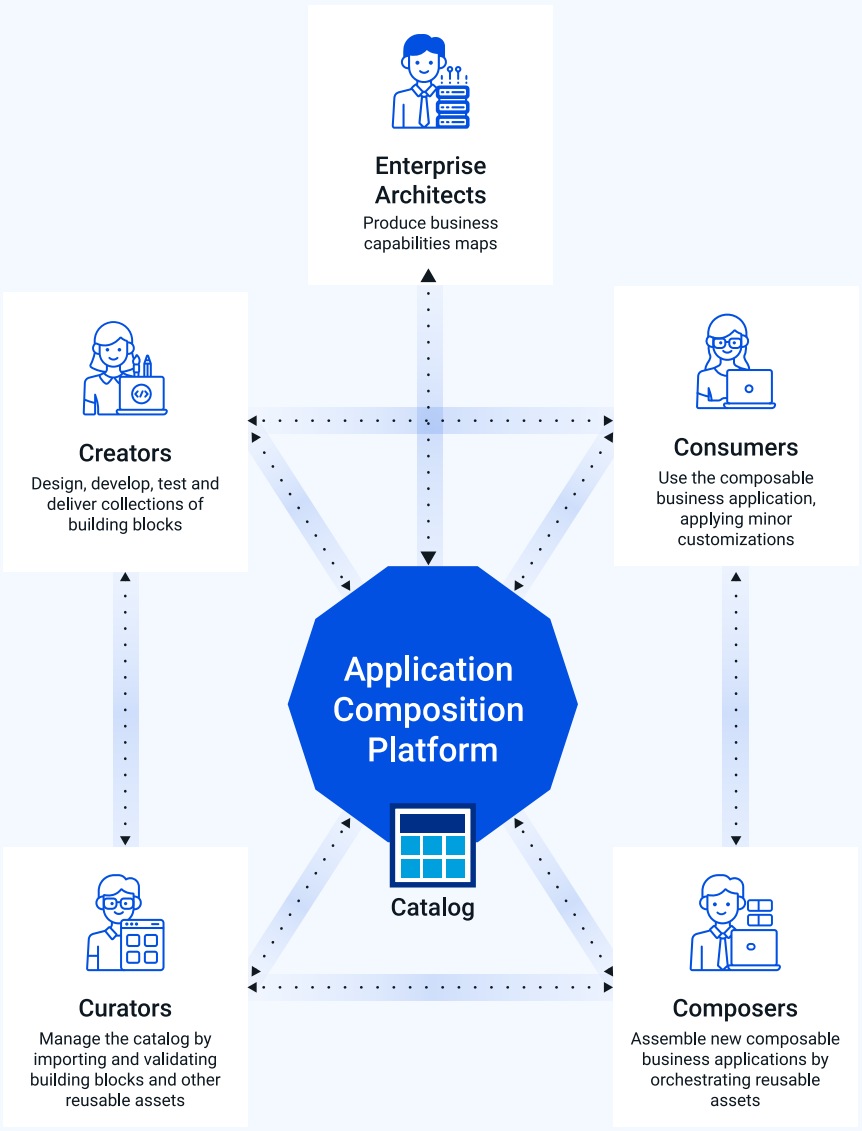


Figure 2 – Building Composable Applications via Business/IT Collaboration

These contributors may or may not all belong to the same team (e.g., curators usually support all the composition teams, but do not report into any). Still, all the roles must work together to successfully deliver, maintain and continually optimize the applications.

4.2 An Emerging Approach to Composition: The Fusion Team

As previously mentioned, one of the key benefits of composable applications is greater alignment between business and IT. If your systems engineers work closely with the business teams, even delegating some aspects of composition to business technologists, the collaboration usually leads to better outcomes. Functional and frontend building block identification and definition, new composable application design, application UX construction, and documentation of reusable assets are all activities that would greatly benefit from a closer business/IT cooperation.

An emerging organizational vehicle that supports such collaborations is the fusion team, sometimes referred to as digital product team, digital delivery team or digital pod. No matter the name, this is a group of business and IT experts collaboratively working together to implement and maintain new composable applications over their lifecycle (see figure 3).

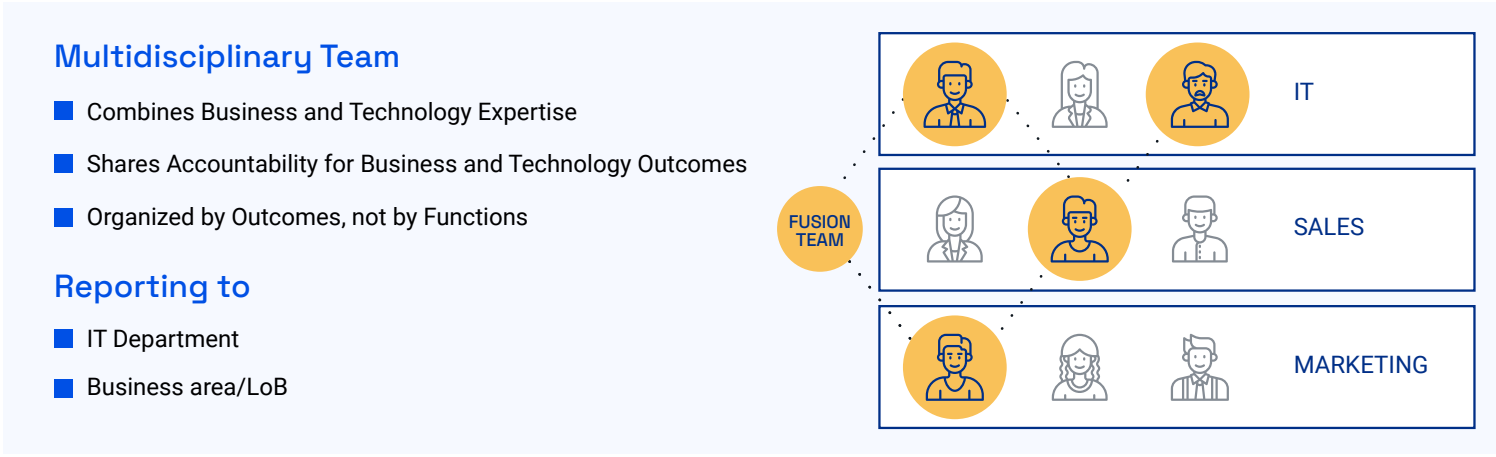


Figure 3 – Fusion Teams as the Composable Applications Organizational Vehicle

You don't necessarily have to structure your composition teams according to these principles. You can adopt a classic approach where the business side formalizes its requirements, and the IT department delivers the composable application that addresses them. However, it is worth noting that the most leading-edge organizations are increasingly endorsing fusion teams as the primary organizational model to support application composition because of its agility and benefits.

In these leading-edge organizations, a fusion team is not a temporary "task force", but a permanent organizational entity aligned with a particular business outcome (for example, digital commerce). Depending on organizational or convenience considerations, a fusion team may report to either central IT, a business team such as sales or marketing, or to another business-oriented entity. The essence is that this multidisciplinary team of business and IT experts exclusively aims to deliver and maintain a composable application (for example, a new B2C digital commerce system) as a "product".

The fusion team is fully accountable from both the technology and business perspectives. If there is a technical problem (e.g., performance, scalability, security or a bug) in the composable application, the relevant fusion team must fix it. If the business initiative enabled by the application doesn't deliver the expected outcomes, the fusion team in charge must find a remedy.

CHAPTER

05

The Application Composition Platform

To provide contributors with all the capabilities they need, your application composition technology foundation must provide an extensive set of functionalities, which we can think of in terms of four macro layers (see figure 4):

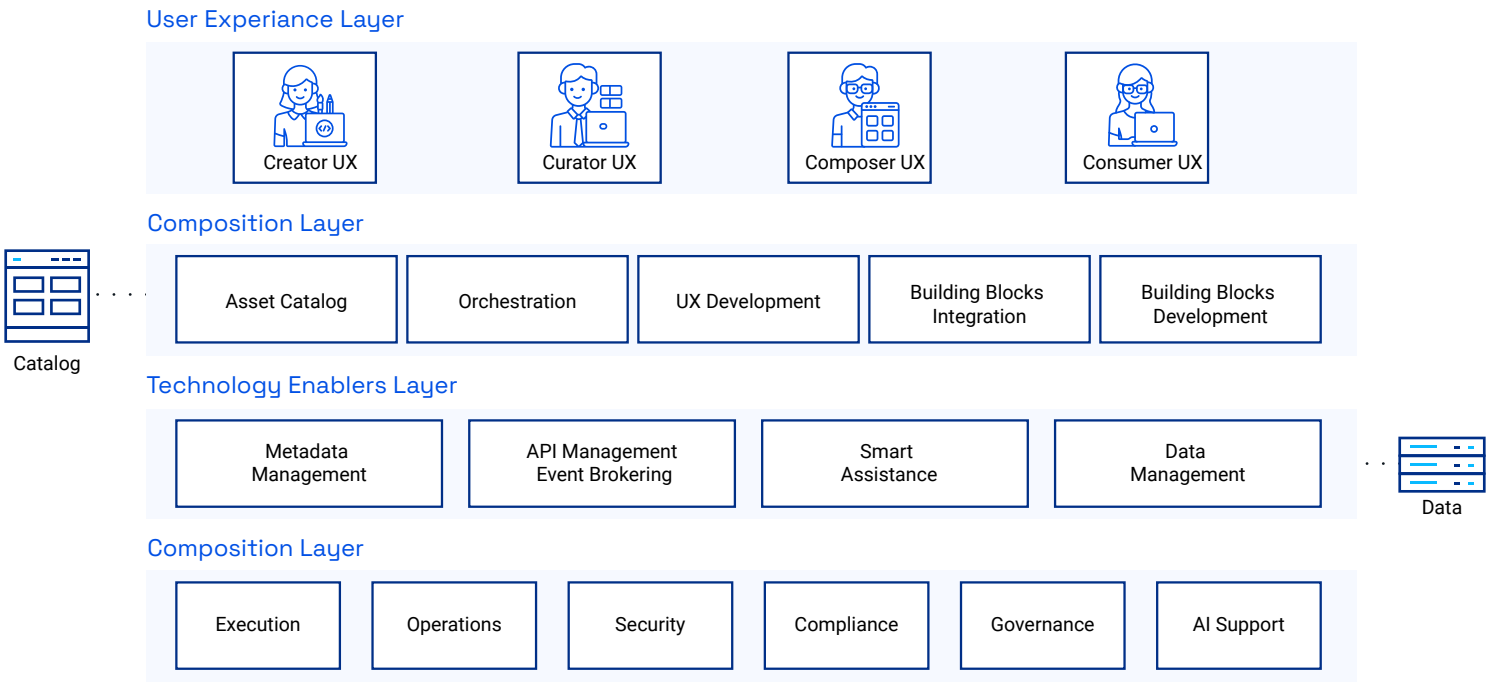


Figure 4 – The Application Composition Technology Foundation

The composition and development layer includes the functionalities used by contributors to build, manage, govern and consume the composable applications (see Note 1).

The operational layer addresses security, compliance, operations, governance, and AI-powered smart assistance requirements (see Note 3).

The technology enabler layer consists of capabilities required for the discovery, interoperability, access to data, deployment, and execution of the various assets in the composable application (see Note 2).

The User Experience Layer enables contributors to access the functionalities of the lower three layers through differentiated, yet collaborative user experiences, tailored to their specific roles and skills as shown in Figure 4.

You could implement the capabilities for your composable application technology foundation by aggregating a range of different tools, some of which have been in the market for many years. Low code application platforms, integration platform as a service, digital experience platforms, business process automation platforms, metadata management tools, API management platforms and event-brokers are examples of products that offer some of the capabilities needed for a foundation, but none of them provides everything you need.

Some trailblazing organizations initially built composable applications on a technology foundation that combined many of these products. However, this is a complex and expensive proposition which takes a long time to implement, and mandates significant effort to operate, govern and maintain. It requires advanced engineering skills that only the most technically astute and wealthy organizations can afford. Practically speaking, composable applications have been an architecture “for the few” until recently, which only deep-pocketed and skill-rich organizations could afford to tackle.

But now, even if you don’t work for one of these organizations, you can venture into composable applications thanks to a new breed of application composition platforms (ACPs) emerging in the market. ACPs are software products and/or cloud services that provide all or most of the capabilities discussed above. You don’t need to buy six or seven products, from different vendors, stitch them together and hope for the best. By adopting an ACP, you have a pre-built technology foundation, ready for your composition teams to use straight out-of-the-box (see figure 5).

Figure 5 – The Application Composition Platform Role in the Composable Applications Landscape

5.1 Application Composition Platform Benefits

What are the benefits of an ACP when compared to a best-of-breed technology foundation that aggregates multiple products?

Simplified and reduced time to deployment

You must purchase, install, configure, test and deploy only one product instead of multiple.

Reduced licensing/subscription costs

In principle, buying a single product is cheaper than procuring an equivalent combination of discrete products, possibly from different vendors. Moreover, a combination of products typically provides you with a superset of functionalities that you pay for, some of which you may never use.

Lower operating costs

Administering, managing, and monitoring a single platform is less expensive than operating a functionally equivalent multivendor combination of products, each with its own administration, management, and monitoring tools.

Greater contributor productivity and lower maintenance cost

If all the contributors use the same platform, designing, composing and maintaining an application is faster and cheaper than building components on different products and then combining assets with different runtimes to seamlessly work together. Moreover, most ACPs come to market with a catalog of out-of-the-box building blocks and other reusable assets provided by the ACP provider or its partners. This further accelerates time to value and reduces development costs for new composable applications.

Shorter learning curve

Contributors only need to learn how to work with a single product, not a plethora of different, likely inconsistent, tools.

Economies of skills

You need fewer engineers to operate and maintain the platform.

5.2 How Do You Choose an Application Composition Platform?

Not surprisingly, all ACPs are not created equal. When choosing an ACP, it is important to carefully evaluate the different alternatives based on the following seven criteria.

Functional completeness:

Does the ACP provide all the capabilities described previously, or at least the subset that is relevant for you? If it doesn't, you may need to extend the ACP with third party products (e.g., an orchestration tool), which will inevitably lead you to face some of the challenges associated with a piecemeal technology foundation.

Architectural consistency:

To what extent are the ACP capabilities truly based on a consistent architecture? Some vendors are coming to market with packages combining preexisting point products into commercial suites, dubbed "application composition platforms." There might be commercial benefits in these suites, but rarely do they have the same degree of consistency and cohesiveness as a "native" ACP, which is built from the ground up as an integrated set of functionalities. If the ACP is just a suite of loosely integrated tools, many of the ACP benefits simply won't materialize.

Plug-and-play modularity:

It would be quite incongruous for an ACP not to be itself composable. In other words, an ACP should ideally allow you to buy and deploy only the functionalities that you need, when you need them. For example, initially, you may not need the integration functionality because you do not expect to use any of the legacy monolithic application's functionality. You plan to develop everything from scratch. Later, however, you realize that you need to use some of the legacy functionality and data for a new project, wrapping it into reusable functional building blocks. Then you can independently buy and deploy the integration functionality in your ACP, which is only possible if your ACP supports plug-and-play modularity.

Modern cloud-native architecture:

The microservices principles of modularity, discoverability, independence and reusability coincide nicely with the tenets of composable application architecture. In principle, you don't need to implement your functional building blocks as microservices. However, most IT leaders will reasonably prefer to invest in a cloud-native, container-based microservices environment to benefit from the intrinsic scalability, reliability, performance benefits, and DevOps and CI/CD support characteristics of such an architecture. Therefore, you may want to look for an ACP based on a cloud-native architecture.

Collaborative composition environment:

As we have seen, contributors applying composable methods use the ACP via a UX tailored to their skills and needs. However, they must all be able to collaboratively share and reuse these assets. For example:

- To assemble a new composable application, composers must be able to reuse building blocks developed by creators and other composers, including themselves.
- Curators must be able to govern and manage every kind of asset, regardless of who they were developed by or which team they came from.
- Consumers must be able to use the composable applications no matter which team developed them.

Moreover, all contributors must be able to exchange feedback and requirements, and otherwise participate in the asset's lifecycle management at the appropriate times, depending on their roles. In a nutshell, an ACP should not only provide a multi-persona UX, it should also enable different contributors to work collaboratively in the context of the composition team they belong to, and potentially, across different teams.

Hybrid, multi-cloud deployment model:

Although ACPs should preferably be cloud-native, it doesn't mean they should only be available as cloud services. Certain composable applications or building blocks may need to run on your organization's private cloud or a traditional data center. An ACP should allow for such a hybrid, multi-cloud deployment model, so that your composable applications can run either fully or in part, in your on-premises data center, in one or multiple clouds, or in any combination of these environments.

Ecosystem:

This is a highly important attribute for an ACP, and its success or failure will be largely determined by the support provided by its ecosystem. The expectation is that the ACP ecosystem offers a rich set of predefined assets, which your teams can leverage to build new composable applications quickly. While ACP providers will develop a certain amount of these predefined assets, the most successful ones will be those that can attract ISVs and service providers investments. These firms will build proprietary assets and make them available to the ACP's customers via the platform catalog (this is why some vendors prefer to call it a "marketplace" or "hub").

Notes

Note 1 - The Development and Composition Layer

The functionalities in this layer are leveraged by all contributors, and includes the following:

- **Asset catalog** — Offers contributors a centralized repository where they can gather, monitor, and access a wide range of reusable assets, including functional and frontend building blocks, API definitions, event schemas, orchestration models, data schemas, fully backed composable applications, and more. In the catalog, these assets are organized into logical domains, facilitating the establishment and enforcement of security measures and usage restrictions. Additionally, it enables integrated asset discovery, orchestration, development, tracking, support, governance, and lifecycle management.
- **Orchestration** — Enables composers to design, implement, deploy, run, and maintain multi-step orchestrations to assemble building blocks into new composable applications.
- **Building block development** — Enables creators to design, implement, deploy, run and maintain functional and frontend building blocks by using pro-code and/or low-code tools.
- **Building block integration** — Enables creators to design, implement, deploy, run, and maintain assets by wrapping legacy applications in APIs or event channels, so that composers can reuse them for building new composable applications. The integration functionality is also used by creators to wrap third party building blocks in APIs and event channels that comply with an organization's standards.
- **User experience development** — Enables creators and composers to design, implement, deploy and run omnichannel digital user experiences (UXs) that aggregate multiple frontend building blocks. These digital UXs engage with the appropriate functional building blocks via APIs and event channels that those building blocks expose.

Note 2 - The Technology Enabler Layer

The functionalities in this layer primarily target creators and curators, and consists of the following capabilities:

- **Metadata management** — Enables contributors to store, retrieve and track metadata that describes, associates, registers, and monitors the state of the different types of reusable assets. This technology is the key enabler of the component catalog.
- **API management and event brokering** — Enables contributors to define, implement and monitor APIs and event channels and manage their lifecycle, orchestrate the associated services, transform payloads, and leverage and bridge relevant transport protocols and data formats.
- **Smart Assistance** — Uses ML, NLP, chatbot, generative AI, and other techniques to intelligently assist contributors in developing, deploying, running, operating, scaling, troubleshooting, and managing composable applications to improve productivity, reduce skills demand, and automate operations.
- **Data management** — Enables contributors to store data in different formats, manipulate them, and manage their quality.

Note 3 - The Operational Layer

The functionalities in this layer primarily target curators, typically in their "technology operations" role. However, all other contributors benefit indirectly as operations directly impact the composition and technology enabler layers. The operational layer is the composable application's "operating system" that provides these services:

- **Execution** – This functionality provides a high performance, highly available, and disaster-recoverable runtime environment for composable applications.
- **Operations** — Includes capabilities required to deploy, provision, administer, monitor, and manage the technology foundation, the composable applications running on it, and their component parts.
- **Security** — Prevents malicious, fraudulent, or undesired access to the composable applications, their component parts, their data, and the foundation technology layers themselves through properly defined policies.
- **Compliance** — Similar to security, it ensures policy-based compliance to corporate and governmental standards.
- **Governance** — This catalog-driven functionality supports governance of the composite applications and their component parts. It includes policy management and enforcement, lifecycle management, execution tracking and operational analytics for composition technology activities.
- **AI support** — Provides the AI functionalities needed to implement the smart assistance capability in the technology enabler layer. Some of these capabilities are also typically made available to creators and composers to make it possible for them to inject AI-driven functionality in the building blocks

CHAPTER

06

How to Get Started with Composable Applications

As an IT leader, you cannot ignore composable applications any longer. Over the next five years and beyond, their impact will be as profound as digital technology has been over the past five years. Composable applications will enable your organization to tackle oncoming business challenges with unprecedented agility.

But where do you start? How do you introduce composable applications while maximizing the chances of success and minimizing the risks?

These are objectively inevitable challenges that should not be ignored. However, if the ACP is chosen properly, its business benefits will quickly offset these risks. A question It leaders often ask themselvess is "will this vendor's ACP still be around in five years?" but this a wrong question. The right one to ask should be "how long will it take me to recover the costs of adopting this ACP in terms of business benefits?" If this time is acceptably short, after a few years, you will be able to justify the replacement of the initial ACP with an alternative, if needed, as the original ACP has already recouped its cost.

Therefore, you should focus on selecting an ACP that can fulfill your short (1-2 years) and medium term (3-5 years) requirements. Looking for an ACP for the long term (5-10+ years) is quite difficult at this stage. Relying on bigger stable vendors is not without risk, as they may drop their ACP or replace it with a successful pure player's product, if revenues are not satisfactory. They can even decide to abandon the market altogether.

Build up multidisciplinary skills and competencies about composition in a stepwise fashion by identifying new initiatives that would benefit from a much closer business and IT alignment, possibly in the context of a fusion team. Good examples may be applications meant to support business innovation, requiring short time to value, and expected to change frequently to meet evolving business requirements. To minimize risk, choose an already budgeted application, with the right amount of technical complexity. This way you will test the technology in a reasonably business critical scenario, and your IT and business teams can move up the composable application learning curve in incremental steps. To prove the business value, the selected application should also provide significant and measurable business benefits in a relatively short time, say within six to nine months.

Adopt an ACP, rather than a best-of-breed technology set. While ACPs may be relatively new in the market, they will propel your organization to the composable application approach while reducing risks, optimizing costs, accelerating time to value, fostering distributed business/IT collaborations, and enforcing centralized operations and governance.

Counter vendor and technology viability risks by selecting a fast-return ACP. Choosing a product in the new ACP market is a challenging task, exposing you to risks related to both vendor and technology viability. The vendor may go out of business, be acquired or change direction, drop the ACP due to insufficient revenues, or reposition the product in a different market.

Contrary to intuition, opting for a well-funded and established start-up as your ACP provider could paradoxically offer a safer choice compared to a large vendor. The reason being that for the start-up, the company's future is inextricably tied to the success of their ACP. For a large provider, the ACP investment is a rounding error in their budget and can be abandoned at any time.

Carefully evaluate the parameters of potential ACPs. When choosing an ACP, it is critical for you to evaluate the contenders in terms of the seven criteria discussed in section 5.2, keeping in mind that some criteria may be more relevant than others in your case. For example, you may not care much about the deployment model because your strategy is aligned to a specific cloud hyperscaler. However, offerings that meet all seven criteria are likely to be amongst the "winners" in the race for ACP market domination. Therefore, committing to an ACP that doesn't support these criteria would expose you to serious technology or vendor viability risks.

CHAPTER

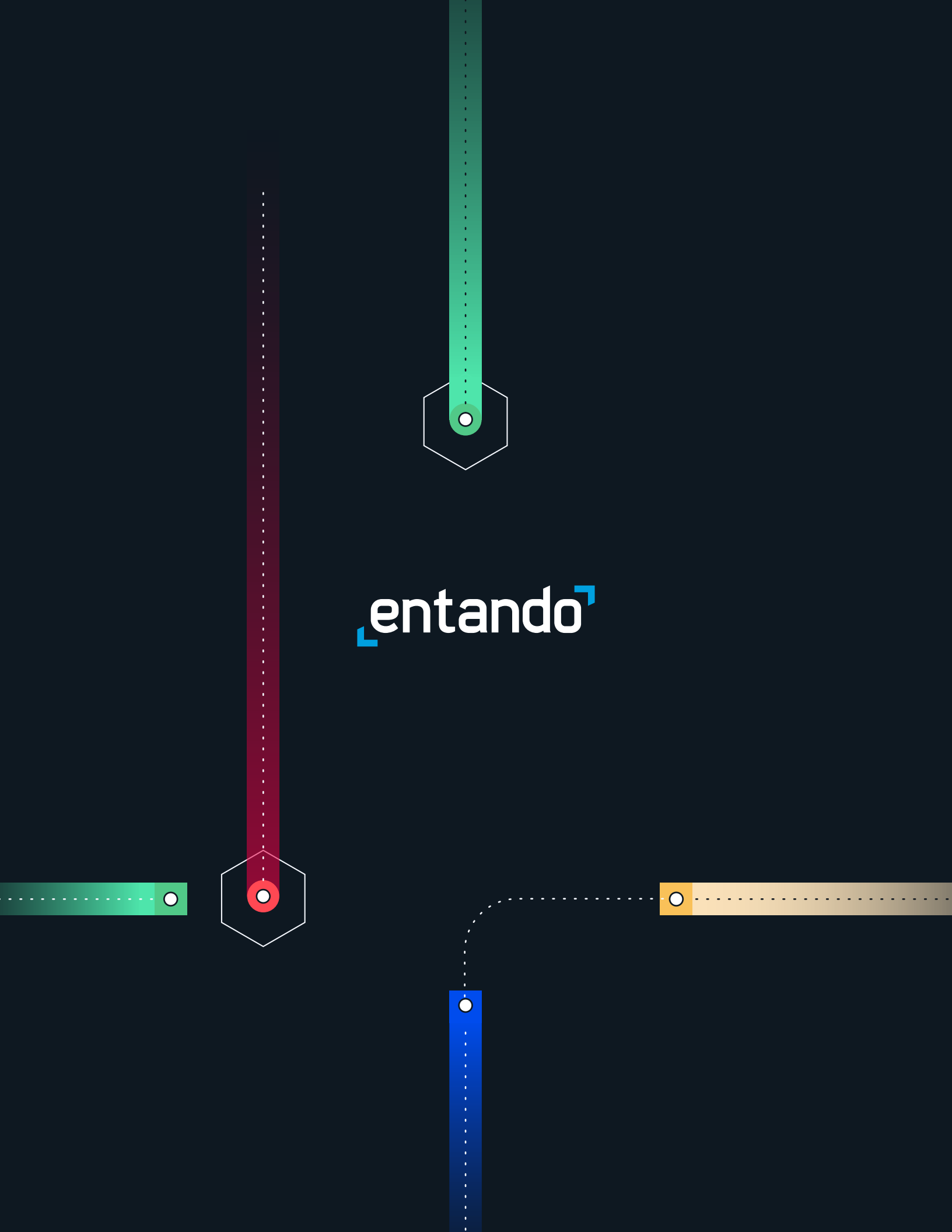
07

Conclusions

There is a composable application in your future! Probably more than one. No matter the size of your organization, which country or vertical sector it operates in, the probability you will have to deal with a composable application is very high—almost a certainty.

Adopting the composable application approach is necessary for your organization to empower its ambitions for greater business agility and responsiveness, shorter time to value for new products and services, and faster innovation cycles whilst keeping IT costs under control.

As an IT leader, you should facilitate composable application adoption by initiating a process to build awareness about the business benefits of composable applications (e.g., via a POC or POV) and, if possible, experimenting with fusion teams to foster synergistic business/IT collaborations. Finally, to maximize positive business outcomes and minimize risks, you should implement a composable technology foundation by selecting a fast-return ACP that meets your short- and medium-term business needs based on the seven evaluation criteria discussed in this article.


entando