IBM | Nube

Search

IBM Cloud Learn Hub > SOA (Arquitectura Orientada a Servicios)

SOA (Arquitectura Orientada a Servicios)

Integración

17 julio 2019

Por: IBM Cloud Education

Explore SOA (arquitectura orientada a servicios), una etapa importante en la evolución del desarrollo y la integración de aplicaciones.

Tabla de contenido

¿Qué es SOA (arquitectura orientada a servicios)?

SOA, o arquitectura orientada a servicios, define una forma de hacer que los com software sean reutilizables a través de interfaces de servicio. Estas interfaces ut

Hablemos

estándares de comunicación comunes de tal manera que pueden incorporarse rápidamente a nuevas aplicaciones sin tener que realizar una integración profunda cada vez.

Cada servicio en una SOA incorpora el código y las integraciones de datos requeridas para ejecutar una función comercial completa y discreta (por ejemplo, verificar el crédito de un cliente, calcular un pago mensual del préstamo o procesar una solicitud de hipoteca). Las interfaces de servicio proporcionan un acoplamiento flexible, lo que significa que pueden llamarse con poco o ningún conocimiento de cómo se implementa la integración debajo. Los servicios se exponen utilizando protocolos de red estándar, como SOAP (protocolo simple de acceso a objetos) / HTTP o JSON / HTTP, para enviar solicitudes para leer o cambiar datos. Los servicios se publican de una manera que permite a los desarrolladores encontrarlos rápidamente y reutilizarlos para ensamblar nuevas aplicaciones.

Estos servicios se pueden construir desde cero, pero a menudo se crean exponiendo funciones de sistemas de registro heredados como interfaces de servicio.

De esta manera, SOA representa una etapa importante en la evolución del desarrollo y la integración de aplicaciones en las últimas décadas. Antes de que SOA surgiera a fines de la década de 1990, la conexión de una aplicación a datos o funcionalidades alojadas en otro sistema requería una integración compleja punto a punto, integración que los desarrolladores tuvieron que recrear, en parte o en su totalidad, para cada nuevo proyecto de desarrollo. Exponer esas funciones a través de SOA elimina la necesidad de recrear la integración profunda cada vez.

Tenga en cuenta que aunque SOA y la arquitectura de microservicios más reciente comparten muchas palabras en común, solo están poco relacionadas y, de hecho, operan en diferentes ámbitos, como se discute más adelante en este artículo.

¿Qué es un ESB?

An ESB, or enterprise service bus, is a pattern whereby a centralized component performs the integration to the backend systems and then makes those integrations available as service interfaces. It performs translation of data models, deep connectivity, routing, and potentially composition of multiple requests, and it makes these available as a single service interface for reuse by new applications. The ESB pattern is typically implemented using a specially designed integration runtime and tooling that are well-suited to the above capabilities, ensuring the best possible productivity.

In theory, you could implement an SOA without an ESB, but application owners have to find their own unique way to expose service interfaces, which is a lot of w the interfaces are eventually reusable) and creates a significant maintenance challenge in the future. In fact, ESBs were, eventually, considered such a de facto element of any SOA implementation that the two terms are sometimes used synonymously, creating confusion.

Learn more about ESBs by reading "Introduction to ESB (Enterprise Service Bus)."

Benefits of SOA

Compared to the architectures that preceded it, SOA offers significant benefits to the enterprise:

- **Greater business agility; faster time to market:** The efficiency of assembling applications from reusable service interfaces, rather than rewriting and reintegrating with every new development project, enables developers to build applications much more quickly in response to new business opportunities.
- **Ability to leverage legacy functionality in new markets:** A well-crafted SOA enables developers to easily take functionality 'locked' in one computing platform or environment and extend it to new environments and markets. For example, many companies have used SOA to expose functionality from mainframe-based financial systems to the web, enabling their customers to serve themselves to processes and information previously accessible only through direct interaction with the company's employees or business partners.
- **Improved collaboration between business and IT:** In an SOA, services can be defined in business terms (e.g., 'generate insurance quote' or 'calculate capital equipment ROI'). This enables business analysts to work more effectively with developers on important insights—such as the scope of a business process defined by a service or the business implications of changing a process—that can lead to a better result.

SOA Examples

By 2010, SOA implementations were going full steam at leading companies in virtually every industry. For example:

- Delaware Electric turned to SOA to integrate systems that previously did not talk to each other, resulting in development efficiencies that helped the organization stay solvent during a five-year, state-mandated freeze on electric rates.

- Cisco adopted SOA to make sure its product ordering experience was consistent across all products and channels by exposing ordering processes as services that Cisco's divisions, acquisitions, and business partners could incorporate into their web sites.
- Independence Blue Cross (IBC) of Philadelphia implemented an SOA to ensure that the different constituents dealing with patient data—IBC customer service agents, physicians' offices, IBC web site users—were working with the same data source (a 'single version of the truth').

You can read more details about these and several other SOA use cases in *Service Oriented Architecture for Dummies* (available at no charge, no registration required).

SOA vs. microservices

Experts have filled a few thousand of print and digital pages comparing SOA and microservices and defining the subtleties of their relationship to one another. For the purposes of this article, the chief differences between the two are the coupling of components and scope of use:

- SOA is an enterprise-wide concept. It enables existing applications to be exposed over loosely-coupled interfaces, each corresponding to a business function, that enables applications in one part of an extended enterprise to reuse functionality in other applications.
- Microservices architecture is an application-scoped concept. It enables the internals of a single application to be broken up into small pieces that can be independently changed, scaled, and administered. It does not define how applications talk to one another—for that we are back to the enterprise scope of the service interfaces provided by SOA.

Microservices architecture emerged and gained steam with the rises of virtualization, cloud computing, Agile development practices, and DevOps. Most of the advantages of microservices in these contexts arise from the complete decoupling of the components, which simplifies and improves the following:

- **Developer agility and productivity:** Microservices enable developers to incorporate new technologies to one part of the application without catching up the rest of the application. Any component can be modified, tested, and deployed independently of the others, which speeds iteration cycles.
- **Scalability:** Microservices can take maximum advantage of cloud scalability- component can be scaled independently of the others for the fastest possibl workload demands and the most efficient use of computing resources.

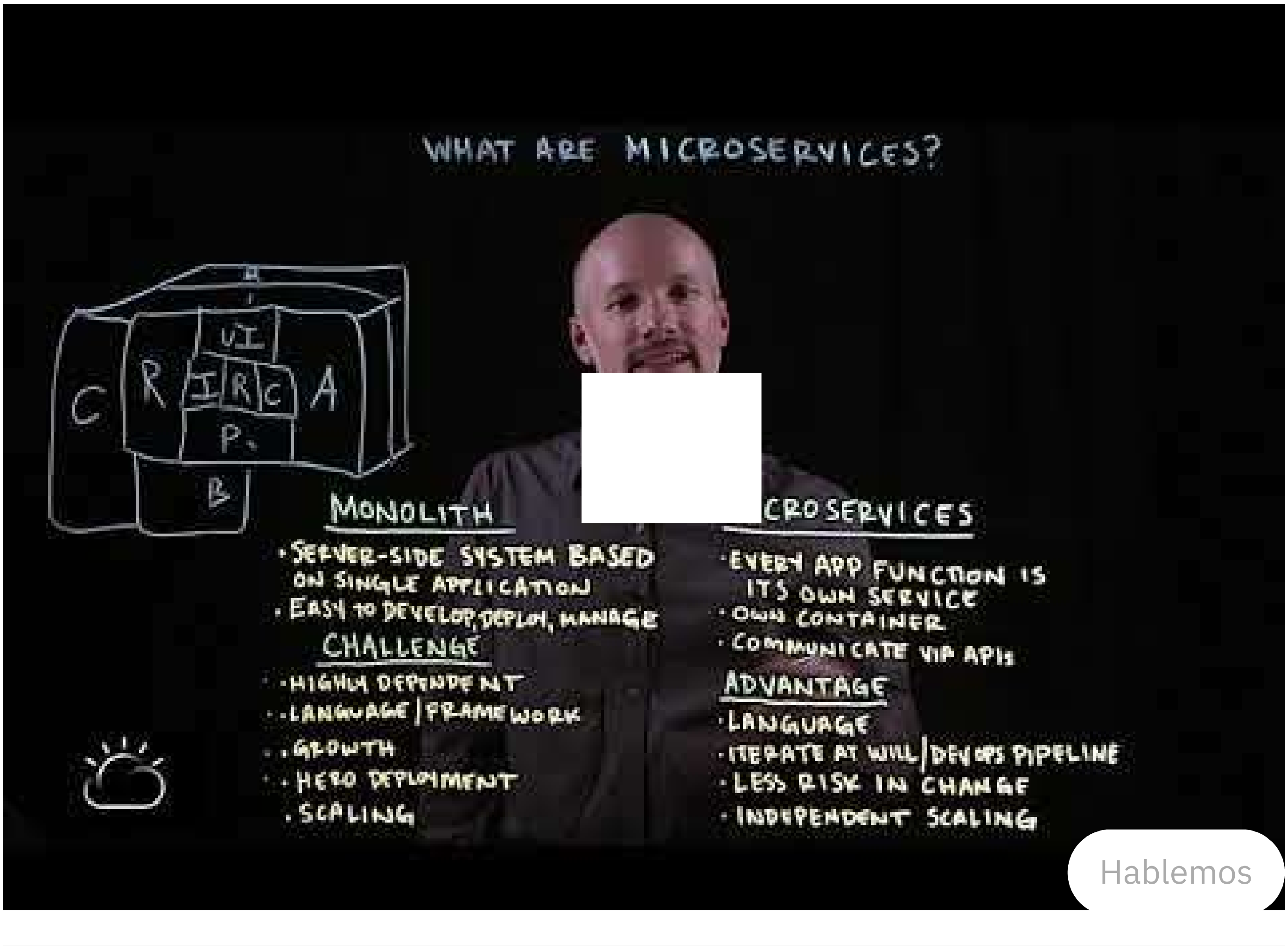
workload demands and the most efficient use of computing resources.

- **Resilience:** Again, thanks to decoupling, the failure of one microservice does not impact the others. And each microservice can perform to its own availability requirements without staking the other components or the entire application to greatest common availability requirements.

For a deeper dive into the differences between SOA and microservices, including their respective places in the evolution of application development and integration, and how they can potentially complement each other within your IT infrastructure, explore these resources.

In the same way that microservices architecture has the potential to bring improvements in agility, scalability, and resilience to application design, these same techniques can also be applied to integration. This is important because, over time, the heavily centralized ESB pattern and its associated centralized team of integration specialists can become a bottleneck. Borrowing from microservices principles, we can potentially break up the ESB into a more fine-grained, decentralized integrations. This is one of the core premises behind agile integration.

For more of an explanation of microservices architectures, see the video “What are Microservices?”:



What are Microservices?

SOA and IBM Cloud

A medida que su empresa cambia su infraestructura de TI hacia un enfoque de nube híbrida , existe una alta probabilidad de que trabaje con una variedad de cargas de trabajo, incluidas aquellas basadas en SOA. IBM es uno de los pioneros de SOA, y las ofertas y servicios de IBM Cloud pueden aprovechar y ampliar sus inversiones SOA existentes.

Visite la página de inicio de IBM Cloud Integration para aprender cómo puede conectar todas sus aplicaciones y datos a través de múltiples nubes privadas y públicas para crear experiencias de clientes personalizadas.

Comience hoy mismo con una cuenta de IBM Cloud .

Siga IBM Cloud

IBM Cloud News lo conecta con el conocimiento y la información que puede poner a trabajar de inmediato, directamente de la mente de los expertos de IBM Cloud, los clientes de IBM y los líderes empresariales y de TI.

Correo electrónico suscribirse
RSS

Productos relacionados

App Connect

Todos los temas

Hablemos

Usted también podría estar interesado en

Nube

Cómo asegurar las API REST con certificados de cliente en IBM Cloud

Por: Kohji Ohsawa

6 de febrero de 2020 →

Automatización

Cómo programar llamadas de API REST en IBM Cloud

Por: Gabriele de Capoa, Crescenzo Migliaccio y Emilia Amato

30 de enero de 2020 →

Integra

Uso

Senc

Por: Ra

28 de e

Sea el primero en escuchar noticias, actualizaciones de productos e innovación de IBM Cloud

Recibe actualizaciones en tu bandeja de entrada. →

Aprender

¿Qué es la computación en la nube?

¿Qué es Kubernetes?

¿Qué son los contenedores?

¿Qué es la informática sin servidor?

¿Qué es DevOps?

Centros de datos

Hablemos

Productos

Soluciones

Clientes

Seguridad

Construir

Empezar

Docs

Cursos

Practicas

Arquitecturas

Cadenas de herramientas

Contratar

Apoyo

Servicios

Garaje

Socios

Blog

Boletin informativo

Gorjeo

Youtube

Facebook

Su cuenta

Regístrate

Iniciar sesión

Consigue ayuda

Problemas para entrar

Hablemos

Hablemos