***SOA***

As arquiteturas orientadas a serviços (SOA, do inglês service-oriented architetures) são uma forma de desenvolvimento de **sistemas distribuídos em que os componentes de sistema são serviços autônomos, executando em computadores geograficamente distribuídos. Protocolos-padrão baseados em XML, SOAP e WSDL foram projetados para oferecer suporte à comunicação de serviço e à troca de informações.**Consequentemente, os serviços são plataforma e implementação independentes de linguagem. ***Fonte: Engenharia de Software - Ian Sommerville - 9ª Ed. (pág. 356).***

Para que a SOA possa se propagada de forma bem sucedida a maioria dos autores afirma que ela exige governança na organização, não da para implementar SOA sem ter a governança como diretriz. Governança SOA é ampliada pela Governança de TI que deve está alinhado com a Arquitetura de Governança Empresarial que dão suporte à Governança de Negócio como um todo, ou seja, todas devem está alinhadas.

É um conceito abstrato para desenvolvimento de software que visa definir a integração entre os diversos serviços que compõem uma aplicação sem, entretanto, entrar nos méritos de sua implementação. Em resumo, uma arquitetura orientada a serviços é uma arquitetura de sistemas distribuídos tipicamente caracterizada pelas seguintes propriedades [w3c\_wsarch]:

* **Visão lógica**: o serviço é definido em termos funcionais, tipicamente implementando operações de negócios. A visão lógica do serviço o abstrai dos programas, bases de dados, processos de negócios, etc., necessários à sua implementação;
* **Orientado a mensagens**: o serviço é formalmente definido em termos de mensagens trocadas entre unidades de software provedoras e consumidoras do serviço. A estrutura interna das unidades de software, incluindo sua linguagem de programação, sua estrutura de dados, etc. são deliberadamente abstraída na SOA. Este baixo acoplamento entre aplicações provedoras e consumidoras de um serviço traz benefícios significativos para sistemas legados: a transparência da implementação interna das aplicações de uma arquitetura SOA permite que qualquer componente de software ou aplicação seja envolvido por uma interface de mensagens e adaptado a uma definição de serviços formal.
* **Orientado a descrição**: um serviço é descrito por uma meta-linguagem processável por máquinas. A descrição suporta a natureza pública da SOA: apenas os detalhes do serviço relevantes a seu uso são incluídos na descrição;
* **Granularidade**: serviços tendem a utilizar um pequeno número de operações com mensagens relativamente grandes e complexas;
* **Orientado a redes**: serviços tendem a ser disponibilizados e utilizados através de uma rede, embora este não seja um requisito obrigatório;
* **Independência de plataforma**: mensagens são enviadas num formato padrão, independente da plataforma, através de interfaces bem definidas.

Arquitetura Orientada a Serviços (**SOA**) não é uma tecnologia, não é uma metodologia, não é um serviço, mas é um **conceito** de arquitetura corporativo que promove a integração entre o negócio e a TI por meio de conjunto de interfaces de serviços acoplados. Tanto a orquestração quanto a coreografia, no paradigma SOA, arranjam serviços diferentes para serem executados em uma ordem preestabelecida. A diferença é que a orquestração possui um orquestrador único que define a ordem de realização dos serviços, fazendo com que os mesmos não necessitem conhecer uns aos outros. Na coreografia, é necessário que os serviços saibam da existência uns dos outros, para, então, serem executados em uma ordem preestabelecida.

O **XML** é um formato para a criação de documentos com dados organizados de forma hierárquica, como se vê, frequentemente, em documentos de texto formatados, imagens vetoriais ou bancos de dados. <https://pt.wikipedia.org/wiki/XML>.

Segundo Fábio Perez Marzullo, em seu livro SOA Na Prática, inicialmente, quatro **papéis são essenciais** para a composição do **ciclo de vida de soluções SOA**:

* **Consultor de negócios** - mapeia processos de negócio da organização.
* **Arquiteto SOA** - quem modela, constrói, instala e dá manutenção ao SOA.
* **Provedor** - quem disponibiliza serviço.
* **Cliente/consumidor** - quem invoca serviço.

A **Arquitetura Orientada a Serviços** (Service-Oriented Architeture) ou **SOA**, é uma arquitetura de software que se baseia na computação distribuída e cujo princípio é a **entrega de funcionalidades** (serviços) ao invés de sistemas completos. O principal elemento para permitir que aplicações de diferentes sistemas se comuniquem dentro da companhia é através dos **web services**.

Esse é o princípio da arquitetura SOA, lidar com **ambientes heterogêneos** e integrar as **regras de negócio** com o setor tecnológico da companhia, distribuindo seus **serviços** através de um sistema **web service** que pode ser utilizado por diferentes sistemas operacionais e softwares.

**Modelo Triangular x SOA**

* **Provedor de serviços**: cria e desenvolve o serviço Web que serve para expor alguma funcionalidade de negócio da sua organização para a invocação por outros usuários externos. Responsável pela PUBLICAÇÃO do serviço. Também é o responsável por fornecer toda a infraestrutura de acesso e responder a requisições internas e externas.
* **Consumidor de serviços ou cliente**: qualquer usuário que deseja utilizar algum serviço Web. Responsável pela PESQUISA do serviço.
* **Catálogo de Serviços (UDDI)/Registro dos serviços**: um diretório central onde o provedor de serviços possa cadastrar e descrever seus serviços, e onde o consumidor possa pesquisar/procurar pelo serviço desejado. Responsável pela LIGAÇÃO com o serviço.

Três papéis são identificados de acordo com os comportamentos e responsabilidades próprias de um serviço.

* Provedor de Serviço: aquele que oferece o serviço;
* Consumidor de Serviço: aquele que utiliza o serviço;
* O Registro de Serviço: mecanismo que permite ao provedor cadastrar seus serviços e ao consumidor encontrá-los.

Princípios SOA:

* Padronização do contrato de serviço
* Baixo acoplamento
* Abstração do serviço
* Autonomia do serviço
* Visibilidade do serviço
* Independência do controle de estado do serviço
* Reusabilidade
* Capacidade de composição do serviço

**Orquestração:** composição de processos de negócio (por exemplo, por meio de Web  
Services na qual existe a figura de um processo central (processo mestre) que controla e coordena os demais processos). Neste tipo de composição, cada processo participante não tem conhecimento de que faz parte de uma composição de processos, com exceção do processo mestre.

**Coreografia**: composição de processos de negócio (por exemplo, por meio de Web Services na qual não existe a figura de um processo mestre que controla e coordena os demais processos). Neste tipo de composição, cada processo envolvido tem o conhecimento de que faz parte de uma composição de processos e que precisa interagir com outros processos de maneira ordenada para que a composição resultante tenha sucesso.

"Dentro de uma solução orientada a serviços, as **unidades de lógica (serviços) encapsulam funcionalidades não especificas a nenhum aplicativo ou processo de negocio**. Esses serviços são classificados como ativos de TI **agnósticos** e reusáveis. Serviços agnósticos fornecem um intervalo de funcionalidades genéricas."

Maturidade SOA:

* **Nível 1: Silo –**Cada setor da empresa desenvolve suas próprias aplicações de forma independente, sem integração de dados, processos, padrões ou tecnologia.
* **Nível 2: Integrado –**A tecnologia é usada para integrar os silos e os dados, porém é necessária uma conversão de dados e protocolos, uma vez que os sistemas e processos não obedecem a padrões.
* **Nível 3: Componentizado –**Os sistemas são decompostos em componentes, porém a integração entre eles possui um alto acoplamento. Os componentes são reutilizáveis  através de EAI - *Enterprise Application Integration*, mas o ambiente ainda apresenta muitas replicações e redundâncias.
* **Nível 4: Serviço –**Aplicações são construídas através da composição de Serviços. Os Serviços são identificados e construídos com base nos Princípios de *Design* de Serviços, garantindo o baixo acoplamento e a interoperabilidade, independentemente da tecnologia utilizada.
* **Nível 5: Serviços Compostos –**Neste nível a empresa constrói suas aplicações através da interação de vários serviços, usando uma linguagem de modelagem de processos de negócio, como BPEL. Com isto a empresa ganha agilidade na resposta a novas solicitações das áreas de negócio.
* **Nível 6: Serviços Virtualizados -**Os Serviços passam a ser disponibilizados por meio de fachadas (*façades*) de modo que seu acesso seja feito de maneira indireta, permitindo que os Serviços possuam um baixo acoplamento à infraestrutura utilizada. Isto cria mais possibilidades de composição de Serviços.
* **Nível 7: Serviços Reconfigurados Dinamicamente –**Serviços podem ser localizados e compostos dinamicamente, sem necessidade de intervenção de desenvolvedores. Embora possa parecer muito futurista, já existe tecnologia para que isto possa acontecer.

**As principais características dos serviços Web são:**  
  
**Baseado em XML**: usado para representar os dados. Como transporte de dados, XML (eXtensible Markup Language) [4] elimina qualquer dependência com rede e sistema operacional.

**Fracamente acoplado**: a interface de um serviço Web pode mudar durante o tempo sem comprometer a habilidade do cliente de interagir com o serviço.  
**Granularidade grossa**: provê uma maneira natural de definir serviços de granularidade grossa que acessam a quantidade correta de lógica de negócio.  
**Chamadas síncronas e assíncronas**: um cliente pode invocá-lo de forma síncrona e assíncrona. Possibilitar chamadas assíncronas é a chave para permitir sistemas fracamente acoplados.

Qual o relacionamento entre eles (*web services* X SOA)? Embora eles apareçam juntos e o uso deles seja feito de forma conjunta (visto que a forma mais comum de se implementar um SOA é por meio de WS), pode-se ter um webservice isolado e você pode ter uma arquitetura orientada a serviços (SOA) sem webservice;

Os Webservices frequentemente são usados para se implementar uma arquitetura orientada a serviços (SOA), mas não é obrigatório. Ter webservices não implica SOA;

Da mesma forma, ter SOA não implica o uso de Webservices. Ou seja, embora a utilização deles estejam diretamente relacionadas, eles podem ser implementados de forma independente.

**Tipos de serviços usados na SOA:**

* Business services;
* Entity services;
* Functional services;
* Utility services;

**Barramento de Serviços (Enterprise Service BUS - ESB):** modelo conceitual tecnológico que se utiliza de padrões e ferramentas de modelagem e desenvolvimento para unir e conectar serviços, aplicações e recursos de TI da organização.

É um conceito que viabiliza o uso de SOA como infraestrutura de soluções corporativas; regulamenta a forma como os serviços se comunicam e interagem uns com os outros.

**Arquitetura de objetos distribuídos.**

Uma abordagem mais geral para sistemas distribuídos é eliminar a distinção entre cliente e servidor e projetar a arquitetura de sistema como uma arquitetura *de objetos distribuídos. Os componentes do sistema são objetos que oferecem uma interface para um conjunto de serviços que eles fornecem. Outros objetos solicitam esses serviços sem fazer distinção lógica entre um cliente (quem recebe o serviço) e um servidor (quem provê o serviço).*

*Nesse caso pode-se pensar na camada de middleware como um barramento de software.*Ela permite que objetos se comuniquem e sejam acrescentados aos sistemas. Essa arquitetura permite adiar decisões sobre onde ou como os serviços devem ser fornecidos e também permite que novos recursos sejam acrescentados conforme necessário.