Web Services – Fornece uma interface de serviço que permite aos clientes interagirem com servidores de uma maneira mais geral, tornando-a mais rica e estruturada do que acontece com os navegadores web. Incluem uma API que permite aceder a serviços. Os pedidos e respostas são usualmente codificados numa Representação Externa de Dados (RED) em XML e transmitidos sobre HTTP. São identificados por um URI (URL ou URN).

Interface de serviço: Conjunto de operações que pode ser usada por um client na Internet.

URI: Identificador de recurso geral, cujo valor pode ser um URL ou um URN.
URL: Inclui informações de localização do recurso na web, como o nome de domínio do servidor.
URN: Contam com um serviço de pesquisa para fazer o mapeamento para os URLs dos recursos.

Descrição do serviço: Inclui uma definição de interface e outras informações, como o URL do servidor.
A service description pode ser usada para gerar as rotinas de marshalling e unmarshalling de forma automática.

Web Services Description Language (WSDL): Linguagem especifica para descrever serviços web.

Simple Object Access Protocol (SOAP): Protocolo utilizado para a troca de mensagens em sistemas distribuídos que trata do correto encapsulamento dos dados em XML. Em comparação com o XML-RPC, é uma extensão mais avançada e flexível.
Uso de XML em SOAP e nos dados:

- Vantagem: mais legível, por humanos.
- Desvantagem: processamento mais lento que formatos binários.

A arquitetura de Web Services segue um modelo em que o cliente do Web Service poderá consultar a descrição do serviço de nomes ou de diretoria em tempo de execução. Faz uso do protocolo UDDI.

Universal Description Discovery and Integration (UDDI): Protocolo para publicar e pesquisar meta-informação sobre web services. Ele desempenha um papel crucial na arquitetura de web services, permitindo que aplicações descubram dinamicamente e utilizem serviços web em tempo de execução.

SOAP em Java: O JAX-WS é uma API em Java projetada para simplificar o desenvolvimento e consumo de serviços web baseados em XML.

Web Services em Java: Servidor e descrição
Em resumo, o desenvolvimento de um Web Service em Java envolve a criação da Service Interface, a geração automática do Skeleton e da descrição do serviço em WSDL, a execução no Servlet Container e, finalmente, a geração dinâmica do Proxy do cliente. Esses passos simplificam o processo de criação e implementação de serviços web interoperáveis.

Servlet Container: É um servidor que hospeda e executa aplicações web baseadas em Servlets, incluindo serviços web. O dispatcher do Servlet Container é responsável por receber e encaminhar solicitações HTTP para os componentes apropriados da aplicação web.

Segurança no XML: Capacidade de assinar digitalmente ou cifrar o conteúdo. Assinar digitalmente implica em adicionar uma assinatura digital ao documento XML para garantir a sua autenticidade e integridade. Cifrar envolve proteger o conteúdo para que só o destinatário autorizado possa lê-lo. O XML permite várias formas sintáticas de representar os mesmos dados. Quando se aplica uma assinatura digital a um documento XML, é comum preceder esse processo com uma conversão para o Canonical XML. O Canonical XML é uma representação normalizada e serializada do XML
.
GRID Computing: Refere-se ao middleware desenhado para permitir e otimizar a partilha de recursos em larga escala. O principal objetivo é permitir que utilizadores, colaborem eficientemente para alcançar objetivos comuns.
Os recursos envolvidos no Grid Computing estão distribuídos em computadores de diferentes plataformas e ambientes heterogêneos. O middleware usado em sistemas de Grid Computing muitas vezes é projetado para funcionar com base em princípios de serviços web.

GRID vs. Cloud:
Usualmente, os sistemas de Grid Computing são heterogêneos. Grid Computing é especialmente adequado para lidar com enormes volumes de dados ou processamentos complexos. As infraestruturas de Grid Computing geralmente são geograficamente dispersas. O middleware comumente usado em sistemas de Grid Computing é o Globus Toolkit.
Cloud Computing é um paradigma mais recente de computação distribuída de larga escala. é motivada por uma utilização mais geral, sendo capaz de lidar tanto com grandes conjuntos de dados quanto com cargas de trabalho menores. Cloud Computing faz uso extensivo de virtualização.

Cloud computing: Ela permite aos utilizadores aproveitarem recursos computacionais disponibilizados como um serviço através da Internet, reduzindo a dependência de recursos locais e oferecendo flexibilidade, mobilidade e eficiência em termos de custo.

Virtualização: A virtualização é a criação de ambientes virtuais num ambiente físico. Em Cloud Computing, a virtualização frequentemente envolve a criação de Máquinas Virtuais (VMs). Cada VM é uma instância isolada de um sistema operacional que roda em cima de um hypervisor, permitindo que várias VMs coexistam em um único servidor físico. Vantagens: Isolamento, cada

VM é isolada das outras. Flexibilidade, VMs podem ser criadas, modificadas e movidas rapidamente. Eficiência de Recursos, otimiza o uso de recursos fisicos, várias VMs partilham o mesmo hardware.

REST – Representational State Transfer - Estilo arquitetural utilizado para projetar sistemas distribuídos. Na arquitetura REST, os objetos têm um estado representado por dados, e essa representação é transportada por meio de pedidos HTTP. Em comparação com os serviços web baseados em SOAP, os serviços REST são geralmente mais leves e mantêm a legibilidade. Eles podem usar formatos de dados como JSON ou XML, proporcionando flexibilidade na representação dos dados. A arquitetura REST é frequentemente utilizada em ambientes de microsserviços.

Microsserviços: Uma abordagem de desenvolvimento de software que separa as tarefas de um processo complexo num conjunto de serviços e processos autónomos, que comunicam através de APIs, cooperando para a realização de um serviço composto.

Caracteristicas fundamentais do Cloud Computing:
Autoatendimento na Hora e em Função da Necessidade, os utilizadores podem provisionar recursos automaticamente, sem a necessidade de interação humana com cada provedor de serviço. Isso permite que os utilizadores obtenham os recursos de que precisam quando precisam, de forma rápida e eficiente.
Acesso com Grande Largura de Banda, a ideia é que o acesso aos recursos em nuvem seja fácil, independente do tipo de dispositivo ou plataforma que o utilizador esteja utilizando.
Resource Pooling (Carregamento Antecipado de uma Bateria de Recursos), diferentes recursos físicos e virtuais são dinamicamente atribuídos e reatribuídos conforme a demanda do consumidor.
Elasticidade, significa que os utilizadores podem obter e libertar recursos conforme necessário.
Mensurabilidade do Serviço, sistemas em nuvem controlam e otimizam automaticamente o uso de recursos por meio de uma capacidade de medição em algum nível de abstração apropriado ao tipo de serviço.

Elastic computing: Tem a ver com a capacidade de ativar dinamicamente o acesso a recursos computacionais de modo a responder a uma carga variável.

Tipo de Cloud: Publica, privada, comunitária e híbrida.

Benefícios de cloud computing: Recursos partilhados, otimização global do uso de recursos, os recursos podem ser agregados para facilitar aplicações com uso intensivo de dados, o cloud computing elimina ou reduz substancialmente o investimento inicial para uma infraestrtura computacional, redução de custo (pay as you go – os clientes pagam apenas pelos recursos que realmente utilizam), capacidade de responder a situações de carga irregulares, virtualização permite ao utilizador o uso de um ambiente que lhe é familiar, em vez de obrigar ao uso de novas ferramentas e protocolos.

Fatores que contribuem para o sucesso do paradigma de computação em nuvem: Mais capacidade para tirar partido de avanços em software, tecnologias de redes, armazenamento e processamento, através das empresas que prestam serviços em nuvem.
Os recursos estão no âmbito de um só administrative domain (AD). Segurança, gestão, tolerância a falhas, e QoS (qualidade do serviço) tornam-se menos complexas, em comparação com um ecossistema heterogéneo e múltiplos ADs.

Desafios em cloud computing: Disponibilidade do serviço, quando o fornecedor (CSP) não consegue fornecer o serviço pode resultar em períodos de inatividade. Variedade de Serviços e Mobilidade do utilizador, uma vez que um utilizador se acostuma a um CSP, pode ser desafiador mudar para outro. Garantir a confidencialidade de Dados e auditabilidade. Risco de estrangulamento por sobrecarga de dados, pode existir sobrecarga quando há uma grande quantidade de dados sendo transferidos entre o utilizador e a nuvem. Imprevisibilidade do desempenho. Garantir a elasticidade para escalar rapidamente, bem como a gestão eficiente de recursos, são desafios importantes. Segurança e Confidencialidade.

Software-as-a-Service (SaaS): As aplicações são disponibilizadas pelo CSP como um serviço. Utilizador não gere a infraestrutura subjacente nem a fonte do software da aplicação. Não adequado para situações em que os dados não podem ser externalizados (alojados fora da empresa).

Platform-as-a-Service (PaaS): O utilizador da nuvem pode implantar aplicações suas, desde que compatíveis com linguagens de programação e outras ferramentas do fornecedor. Não adequado se a aplicação tem de ser portável.

Infrastructure-as-a-Service (IaaS): Utilizador pode implantar e executar qualquer software. Incluindo aplicações e sistemas operativos. Utilizador não controla a infraestrutura subjacente, mas pode controlar o Sistema Operativo.

Java Bean é um componente de software reutilizável. A intenção deste tipo de dados é a facilidade de uso e a sua multiplicação em containers e outras ferramentas.

POJO é um objeto comum, não sujeito a convenções.

Container: Em java, um container é um mecanismo que aloja e controla componentes.

Virtualização: Pilar do Cloud Computing que simplifica a gestão de recursos físicos relacionados com interpretadores, memória e comunicação.
A virtualização de recursos em nuvem é importante para:

- Segurança, permite isolar serviços que executam na mesma plataforma/hardware;
- Desempenho e fiabilidade, permitindo a migração de aplicações de uma plataforma para outra;
- Desenvolvimento e gestão de recursos oferecidos pelos fornecedores;
- Isolamento ou compartimentação do desempenho.

A virtualização é importante:
- O hardware muda mais rápido que o software;
- Facilidade de portabilidade e migração de código;
- Isolamento de componentes com falha ou atacados.

A criação de ambientes virtuais simula ou emula recursos físicos.
Multiplexing: criar múltiplos objetos a partir de uma só instância de objeto físico.
Agregação: criar um recurso virtual a partir de vários objetos físicos/reais.
Emulação: Formar um objeto virtual, do tipo A, a partir de um objeto físico de tipo diferente B.
Multiplexing e emulação: Um endereço virtual emula um endereço real.

Organização por camadas (Layering): Estratégia divide um sistema em camadas distintas, cada uma com uma responsabilidade especifica.

Instruction Set Architecture (ISA): Na fronteira entre o hardware e software.

Application Binary Interface (ABI): Permite que aplicações e bibliotecas acedam ao hardware.

Application Program Interface (API): Conjunto de instruções disponibilizadas como interface para as aplicações. Permite que aplicações acedam à ISA.

Virtual Machine Monitor (VMM / Hypervisor): Particiona/divide os recursos de um computador para uma ou mais VMs. Permite que vários SOs executem em simultâneo numa só plataforma de hardware.

Classificados num de dois tipos:
- Type 1 hypervisor (VMM de tipo 1): Executa diretamente numa certa plataforma. Permite a execução de máquinas virtuais diretamente no hardware. Um “gest” OS executa no segundo nível acima do hardware.
- Type 2 hypersvisor(VMM de tipo 2): Executa sobre um OS. Um “guest” OS executa no terceiro nível acima do hardware.

O Virtual Machine Monitor (VMM), também conhecido como hypervisor, desempenha várias funções críticas para virtualizar o CPU e a memória em um ambiente de máquinas virtuais.
- Intercepta instruções privilegiadas solicitadas por um SO convidado e assegura a correção e segurança da operação/execução.
- Intercepta interrupts e encaminha-os para o respetivo sistema operativo convidado.
- Controla a gestão da memória virtual.
- Mantém uma tabela shadow page para cada SO convidado, replicando alterações feitas por estes na sua shadow page.
- Monitoriza a carga e desempenho do sistema, tomando medidas corretivas para evitar degradação do desempenho.

Desvantagens/riscos associados à virtualização: VMM Rootkit é um tipo de malware com acesso privilegiado ao sistema. Pode permitir que um SO malicioso separado seja executado clandestinamente.

Containers: Recursos para alojar aplicações. Forma mais leve de virtualização orientada para aplicações. Isolamento entre aplicações, e entre aplicações e SO (cada container tem o seu próprio ambiente isolado, permitindo que várias aplicações coexistam no mesmo host sem interferência). Portabilidade: software, dados, configurações e logs/registos de atividade (altamente portáteis, pois podem ser executados de maneira consistente em diferentes ambientes).

Containers Orchestrations: Prática de gerir e coordenar a implementação, escalabilidade e a manutenção de aplicações baseadas em containers.

VMs vs. Containers
Pontos favoráveis aos Containers:
- Permitem concentrar mais carga computacional na mesma plataforma;
- Requerem menos tempo para arranque de uma solução;
- Flexibilidade;
- Maior facilidade para especificar um ambiente de execução.

Desafios:
- Imprevisibilidade do desempenho;
- Esgotar recursos partilhados.

Aplicações em Cloud
Existem 3 grandes categorias: Stream processing pipelines (aplicações lidam com processamento de dados em tempo real), Bactch processing systems (aplicações que processam grandes volumes de dados em batchs) e Web applications (aplicações voltadas para a interação com o utilizador via web).

Data pipelines: Lêem dados de uma fonte, aplicam uma série de operações e emitem o resultado para um repositório.

MapReduce: É uma estrutura de programação e processamento de dados projetada para facilitar a computação paralela distribuída em grandes conjuntos de dados. A API do MapReduce é projetada para ser simples e fácil de usar. O MapReduce fornece uma estrutura escalável e eficiente para lidar com a computação em larga escala, permitindo o processamento distribuído de grandes conjuntos de dados de maneira eficaz e paralela.

Apache Hadoop é um framework de software de código aberto desenhado para processar grandes conjuntos de dados distribuídos em clusters de computadores. Implementa o modelo MapReduce para processamento paralelo (execução simultânea de várias tarefas num sistema).

Serviço de Nomes: É crucial ter mecanismos para nomear e referenciar diversos recursos. Fornecem informação sobre objetos a partir do seu nome num SD. Exemplos: URL (necessário para aceder um documento na web), Hostname (necessário para identificar uma máquina na rede).

Binding: Associação entre nome e objeto.

Internet Domain Name System (DNS): Serviço de nomes amplamente usado na internet e intranets. O DNS permite registar atributos de objetos em geral (mais usado para computadores e endereços), as organizações podem gerir os seus nomes, qualquer nome pode ser resolvido por qualquer cliente em qualquer localização, boa performance.

1) No contexto de cloud computing, qual o impacto da virtualização na previsibilidade e desempenho do serviço? E no uso de recursos?

2) O que entende de microserviços? Que tecnologias implementadas nas aulas práticas podem ser usadas no contexto de microserviços.

3) Escolha múltipla
Na situação que queres personalizar o SO
Outra sobre aspetos de Software

4) Na escolha de Conteiner ou VM, que aspetos ponderar

5) O que é web service

6) O que entende por resolução de nomes controlado pelo servido

7)escolha multipla sobre VMM
Endpoint RMI