**81** [**Q1118952**](https://www.qconcursos.com/questoes-de-concursos/questoes/8c198127-43) [WebServices ,](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/webservices)[SOAP (Simple Object Access Protocol)](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/soap-simple-object-access-protocol)  **Prova:**[CESPE - 2020 - TJ-PA - Analista Judiciário - Análise de Sistemas (Desenvolvimento)](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2020-tj-pa-analista-judiciario-analise-de-sistemas-desenvolvimento)

Assinale a opção que apresenta o protocolo padrão da indústria, embasado em XML, para projetar e desenvolver web service, independentemente de plataforma e linguagem.

**A** SOAP (*simple object access protocol*)

**B** HTTP (*hypertext transfer protocol*)

**C** FTP (*file transfer protocol*)

**D** SMTP (*simple mail transfer protocol*)

**E** IMAP (*internet message access protocol*)

**Assertiva A**

O protocolo padrão da indústria, embasado em XML é SOAP (*simple object access protocol*)

**http** - Protocolo de transferência de HiperTexto - porta 80 ou 443/SSL TLS

**FTP** - Protocolo de transferência de arquivo - porta 20/21

**SMTP** - Protocolo simples de envio de e-mail - porta 25/587

**IMAP** - protocolo de recebimento e-mail - porta 143

**SOAP** - Simple Object Access Protocol - é um protocolo para troca de informações estruturadas em uma plataforma descentralizada e distribuída.

**82** [**Q1118950**](https://www.qconcursos.com/questoes-de-concursos/questoes/8c125b5a-43) [WebServices](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/webservices)  **Prova:**[CESPE - 2020 - TJ-PA - Analista Judiciário - Análise de Sistemas (Desenvolvimento)](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2020-tj-pa-analista-judiciario-analise-de-sistemas-desenvolvimento)

No desenvolvimento de serviços com a utilização do padrão RESTful, a anotação @produces é usada para

**A** indicar o dado a ser produzido e armazenado por um recurso.

**B** especificar o tipo de dado que um recurso pode produzir e enviar para o cliente em resposta a uma solicitação.

**C** especificar o recurso que irá produzir o dado solicitado por um cliente.

**D** especificar o momento em que o dado solicitado pelo cliente será produzido.

**E** estabelecer controles de acesso aos dados produzidos em função de uma solicitação de um cliente.

**Gabarito: B**

No desenvolvimento de serviços com a utilização do padrão RESTful, a anotação @produces é usada para especificar o tipo de dado que um recurso pode produzir e enviar para o cliente em resposta a uma solicitação.

**@Produces:**determina o tipo de retorno esperado, onde MediaType é a representação do recurso.

Nos blogs a seguir há uma boa explicação de como funciona JAX-RS. Vale a pena a leitura, pois há outras formas das bancas explorarem este assunto:

<http://blog.caelum.com.br/arquitetura-rest-com-java-jax-rs/>

<http://blog.gabrielamorim.com/criando-um-web-service-restful-com-jax-rs/>

Questão sobre JAX-RS

A anotação @Produces é utilizada para informar qual o formato/MediaType (XML, JSON, TEXT etc...) será devolvido ao cliente (GET)

O termo REST é para aplicações que seguem alguns princípios, regras e constraints, criando comunicações do tipo GET, POST, PUT, DELETE e mecanismos de autenticação. Uma aplicação que segue os principios de REST é RESTFul

restrições de uma API RESTful devem garantir que cada requisição das aplicações para a API RESTful contenha toda a informação necessária para o processamento da resposta, sem considerar o estado da sessão.

**RESTful - stateless**

@produces especifica o tipo de dado que um recurso pode produzir e enviar para o cliente em resposta a uma solicitação. o envio é no formato MediaType e o tipo de dados são: XML, JSON, TEXT e outros, o retorno ao cliente é no método GET.

@Produces

*The @Produces annotation specifies the list of the media types produced by a particular API or class. This annotation is optional at both the class and API levels. If it is present at both levels, the method level annotation takes precedence. If the annotation is not present at any level, by default the "text/html" Media Type is returned.*

<https://www.juniper.net/documentation/en_US/junos-space-sdk/18.1/apiref/com.juniper.junos_space.sdk.help/html/reference/spaceannoREST.html>

**83** [**Q1118943**](https://www.qconcursos.com/questoes-de-concursos/questoes/8bf7d93b-43) [WebServices](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/webservices)  **Prova:**[CESPE - 2020 - TJ-PA - Analista Judiciário - Análise de Sistemas (Desenvolvimento)](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2020-tj-pa-analista-judiciario-analise-de-sistemas-desenvolvimento)

Na criação de web service utilizando-se Apache CXF, a exposição de interface para clientes remotos pode ser feita com o uso da classe

**A** ServerFactoryBean.

**B** ServerFactory.

**C** ServiceFactoryBean.

**D** ProxyFactoryBean.

**E** ServerCriator.

**This class helps take a Service and expose as a server side endpoint. If there is no Service, it can create one for you using a ReflectionServiceFactoryBean.**

public class **ServerFactoryBean** extends AbstractWSDLBasedEndpointFactory

For most scenarios you'll want to just have the ServerFactoryBean handle everything for you. In such a case, usage might look like this:

 ServerFactoryBean sf = new ServerFactoryBean();

 sf.setServiceClass(MyService.class);

 sf.setAddress("http://localhost:8080/MyService");

 sf.create();

You can also get more advanced and customize the service factory used:

 ReflectionServiceFactory serviceFactory = new ReflectionServiceFactory();

 serviceFactory.setServiceClass(MyService.class);

 ..

 \/\/ Customize service factory here...

 serviceFactory.setWrapped(false);

 ...

 ServerFactoryBean sf = new ServerFactoryBean();

 sf.setServiceFactory(serviceFactory);

 sf.setAddress("http://localhost:8080/MyService");

 sf.create();

fonte: <https://cxf.apache.org/javadoc/latest/org/apache/cxf/frontend/ServerFactoryBean.html>

**84** [**Q1062701**](https://www.qconcursos.com/questoes-de-concursos/questoes/9d50094c-11) [Padrões de projeto (Design Patterns)](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/padroes-de-projeto-design-patterns)  **Prova:**[CESPE / CEBRASPE - 2019 - TCE-RO - Analista de Tecnologia da Informação - Desenvolvimento de Sistemas](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-cebraspe-2019-tce-ro-analista-de-tecnologia-da-informacao-desenvolvimento-de-sistemas)

Deseja-se implementar uma solução para o desenvolvimento de um *display*, o qual mostra o estado de um objeto X a partir do objeto em si, utilizando-se padrão de projeto. *Displays*alternativos são fornecidos, e a necessidade principal é que todos eles sejam automaticamente notificados e atualizados para refletir a mudança ocorrida quando o estado do objeto mudar.

Para resolver esse problema, é mais adequado o uso do padrão

**A** Observer.

**B** Facade.

**C** Adapter.

**D** Iterator.

**E** Visitor.

**Observer**

*Permite que um objeto, observado, notifique automaticamente todos os objetos vinculados a ele (objetos observadores) respeitando a relação um-para-muitos. A notificação ocorre assim que o estado do objeto observado é atualizado.*

*Fonte*<https://www.thiengo.com.br/padrao-de-projeto-observer>

**Letra A**

BIZU: Vários objetos ficam OBSERVANDO um objeto principal, aguardando uma notificação. ENTÃO,

**Só quem OBSERVER que é notificado.**

The observer design pattern is suitable for distributed push-based notifications, because it supports a clean separation between two different components or application layers, such as a data source (business logic) layer and a user interface (display) layer.

Deseja-se implementar uma solução para o desenvolvimento de um *display*, o qual mostra o estado de um objeto X a partir do objeto em si, utilizando-se padrão de projeto. *Displays*alternativos são fornecidos, e a necessidade principal é que todos eles sejam automaticamente notificados e atualizados para refletir a mudança ocorrida quando o estado do objeto mudar.

BIZU: Vários objetos ficam OBSERVANDO um objeto principal, aguardando uma notificação. ENTÃO,

Só quem OBSERVER que é notificado.

**85** [**Q1057711**](https://www.qconcursos.com/questoes-de-concursos/questoes/fa98e454-0c) [WebServices](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/webservices)  **Prova:**[CESPE - 2019 - MPC-PA - Analista Ministerial – Tecnologia da Informação](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2019-mpc-pa-analista-ministerial-tecnologia-da-informacao)

Quanto a *web services*, o REST

**A** possui uma única URL para identificar as diversas operações executadas por um *web service*.

**B** retorna os dados em somente um formato proprietário, chamado de RESTful.

**C** é uma alternativa ao padrão XML para representar informações.

**D** utiliza um formato baseado em execução de métodos para representar trocas de objetos em JavaScript.

**E** é um estilo arquitetônico para implementar *web services*.

Os erros estão rachurados em **vermelho,**vamos lá:

Gab.: **E)**

A) *possui uma****única******URL****para identificar as diversas operações executadas por um web service.*

**De forma alguma, podem ser implementados vários métodos (VÁRIAS URLS).**

B) *retorna os dados em****somente******um formato proprietário****, chamado de RESTful.*

***Os formatos possíveis para se transitar os dados são XML, JSON e Text Plain.***

***RESTful é o modelo arquitetural e não um formato de dados.***

C) é **uma alternativa ao padrão** **XML** para representar informações.

**Não é uma alternativa ao XML, aliás ele pode transitar dados em formato XML**

D) *utiliza um formato baseado em execução de métodos para representar****trocas de objetos em JavaScript.***

***Os formatos possíveis para se transitar os dados são XML, JSON e Text Plain***

**E) é um estilo arquitetônico para implementar *web services*.**

Representational State Transfer (REST), em português Transferência Representacional de Estado, é um **estilo de arquitetura de software** que define um conjunto de restrições a serem usados para a criação de web services (serviços Web).

**Fonte: wikipedia - texto da alternativa "e"**

**REST**= é um arquitetura de**WEB SERVICES**, que implementa via ***XML, JSON e Text Plain.***

**a)** Quem tem url única é o GraphQL, não REST

**b)** REST costuma retornar JSON, mas pode ser XML, YAML, CSV, ou qualquer coisa

**c)** O JSON é uma alternativa ao XML, não o REST

**d)** Pode usar JSON (JavaScript Object Notation) ou qualquer outro formato. Não existe restrição

**e)** correto

**86** [**Q1057708**](https://www.qconcursos.com/questoes-de-concursos/questoes/fa8f5212-0c) [Padrões de projeto (Design Patterns)](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/padroes-de-projeto-design-patterns)  **Prova:**[CESPE - 2019 - MPC-PA - Analista Ministerial – Tecnologia da Informação](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2019-mpc-pa-analista-ministerial-tecnologia-da-informacao)

Assinale a opção que apresenta os três grupos em que se segmentam os *Design Patterns*.

**A** padrões de criação, padrões estruturais e padrões comportamentais

**B** padrões de análise, padrões estruturais e padrões comportamentais

**C** padrões de criação, padrões de construção e padrões comportamentais

**D** padrões de criação, padrões estruturais e padrões de construção

**E** padrões de análise, padrões de construção e padrões comportamentais

**Design Patterns**- se dividem em: **CRIAÇÃO, COMPORTAMENTO E ESTRUTURAIS**

Mnemonico CCE!

**Padrões de criação**

Abstraem o processo de criação de objetos a partir da instanciação de classes

**Padrões Estruturais**

Tratam da forma como classes e objetos estão organizados para formar estruturas maiores

**Padrões Comportamentais**

Preocupam-se com algoritmos e responsabilidades dos objetos

**87** [**Q1038229**](https://www.qconcursos.com/questoes-de-concursos/questoes/296cdf9a-f4) [SOAP (Simple Object Access Protocol)](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/soap-simple-object-access-protocol)  **Prova:**[CESPE / CEBRASPE - 2019 - TJ-AM - Analista Judiciário - Analista de Sistemas](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-cebraspe-2019-tj-am-analista-judiciario-analista-de-sistemas)

Com relação à interoperabilidade entre sistemas, julgue o item seguinte.

Diferentemente do que ocorre em serviços *web* embasados em SOAP, as solicitações e as respostas dos serviços REST não são empacotadas em envelopes.

SOAP

é um protocolo de transferência de mensagens em formato XML para uso em ambientes distribuídos. O padrão SOAP funciona como um tipo de framework que permite a interoperabilidade entre diversas plataformas com mensagens personalizadas.

Aplicando este padrão em Web Services, geralmente usa-se o WSDL para descrever a estrutura das mensagens SOAP e as ações possíveis em um *endpoint*.

Uma das maiores vantagens disso é que várias linguagens e ferramentas conseguem ler e gerar mensagens facilmente. Várias linguagens de programação permitem a geração de objetos de domínio, Stubs e Skeletons a partir da definição do WSDL, permitindo a comunicação remota via  através de chamadas a métodos remotos, inclusive com argumentos complexos, como se fossem chamadas locais.

O problema desse padrão, é que ele adiciona um *overhead* considerável, tanto por ser em XML quanto por adicionar muitas tags de meta-informação. Além disso, a serialização e desserialização das mensagens pode consumir um tempo considerável.

*REST*

O termo *REST*  **Re**presentational **S**tate **T**ransfer.

Nada mais é que um padrão de arquitetura para criar serviços e disponibilizá-los na Web. Um serviço *RESTful* é simplesmente aquele que realiza a implementação deste padrão.

Ou seja, não há diferenças.

O termo REST surgiu com o Roy Fielding em uma  que escreveu em 2000. Nesta dissertação ele classificava um serviço REST com algumas características. As principais, entendo eu, seriam:

Ou seja, um serviço REST não é aquele que simplesmente manda um Json de um lado para o outro. Existem certas regras que a definem para que ela possa ser mais fácil de ser usada, mais confiável e com melhor performance.

Um exemplo que deixa isto mais claro é quando comparamos dois serviços http onde um segue o mantra do REST e o outro não.

Pessoal, encher comentario com textoes mais confundem q ajudam em alguma coisa. Se sua tecnica eh confundir, vc eh o maior prejudicado, pois tambem vai acabar confuso com tanda %$#@\*. Vamos nos concentrar objetivamente no q a questao pede, q eh a comparacao entre duas tecnologias. Vamos 'a questao:

*Diferentemente do que ocorre em serviços web embasados em SOAP, as solicitações e as respostas dos serviços REST não são empacotadas em envelopes.*

De fato, rest nao envolve envelopamento, pois este adiciona overhead, tornando tudo mais pesado. No rest, os dados sao enviados como tal, em requisicoes/respostas HTTP.

Existe ate uma brincadeira sobre isso com o filme VovoZona. O SOAP seria o personagem como ele eh + o personagem desfarcado de VovoZona, q eh uma senhora com aparencia gorda e pesada, "envelopando" o personagem magro.

No rest, apenas o ator 'e 'enviado' e, por consequencia, menos trabalho eh realizado. Deem uma pesquisada em vovozona + soap + rest e vcs vai entender.

Segundo a tecnica de Nishimura, quando compara tecnologias, geralmente a assertiva eh falsa.

Segundo a tecnica de Nishimura, quando menospreza tecnologias, geralmente a assertiva eh falsa.

E quando compara e menospreza ao mesmo tempo? O q seria?

O CESPE ja se deu conta desse "truque".

poderia-se considerar o protocolo http(s) como um envelope, mas isso seria uma extrapolação e invalidaria a questão.

**88** [**Q1038228**](https://www.qconcursos.com/questoes-de-concursos/questoes/2968d4d5-f4) [SOAP (Simple Object Access Protocol)](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/soap-simple-object-access-protocol)  **Prova:**[CESPE / CEBRASPE - 2019 - TJ-AM - Analista Judiciário - Analista de Sistemas](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-cebraspe-2019-tj-am-analista-judiciario-analista-de-sistemas)

Com relação à interoperabilidade entre sistemas, julgue o item seguinte.

Um dos componentes da mensagem SOAP é o protocolo de transporte, que permite representar chamadas de procedimentos remotos, com seus respectivos valores de retorno.

**GABARITO : ERRADO**

De acordo com ***Sommerville*** ( pág 337 e 338, da 9º ed.)

"... Nas **RPCs** (remote procedure calls, do inglês), um componente **chama outro componente como se fosse um método ou procedimento local**.... Um problema com a abordagem RPC para interação é que o**chamador e o chamado precisam estar disponíveis** **no momento da comunicação** e devem saber como se referir um ao outro....Eles se comunicam através de middleware, que é responsável por garantir que as mensagens sejam passadas para o sistema apropriado."

(pág. 359)

" ... Os serviços se comunicam através de troca de mensagens, e essas mensagens são distribuídas usando protocolos padrão de transporte de Internet, como HTTP e TCP/IP...**Ao contrário dos componentes de software os serviços não usam chamadas de procedimento ou métodos remotos**para acessar a funcionalidade associada a outro serviço...."

" Quando você pretende usar um web service, precisa saber **onde se encontra o serviço** (sua URI) e os detalhes de sua interface. Esses são descritos em uma descrição de serviço em uma linguagem .. chamada **WSDL**. A WSDL define três aspectos de um web service : o que faz o serviço, como ele se comunica e onde encontrar...."

Em <https://www.ibm.com/support/knowledgecenter/pt-br/SSEQTP_9.0.5/com.ibm.websphere.base.doc/ae/cwbs_wsdl.html>

podemos verificar que a definição do protocolo de transporte fica no elemento binding, existente da **WSDL,** e não no SOAP como diz a questão (no exemplo, porém, podemos verificar que o uso de RPC é permitido, mas definido dentro do WSDL).

A **resposta é negativa**, pois **o SOAP não possui um componente que seja o protocolo de transporte**, mas ele utiliza um protocolo de transporte para representar chamadas.

Os elementos do SOAP são os seguintes:

***Envelope***

* É obrigatório, ou seja, toda mensagem possui;
* É considerado o elemento raiz de um documento XML;
* Pode conter declarações de namespaces e atributos adicionais;

**Header**

* É um cabeçalho, esse elemento é facultativo;
* Carrega informações adicionais;
* Se ele for utilizado, deve ser o primeiro elemento do envelope;

**Body**

* É o corpo do documento, é um elemento obrigatório;
* Contém as informações transportadas para o usuário final (playload);
* Pode conter o elemento fault:
* Usado para carregar mensagens de erro e status.

Assim, podemos perceber que o SOAP não possui um protocolo de transporte entre os seus elementos, mas ele utiliza um protocolo de transportes para trocar de informações em um ambiente distribuído e descentralizado.

 Inicialmente o SOAP somente utilizava o protocolo HTTP, mas atualmente ele pode utilizar quaisquer outros protocolos, como TCP ou UDP, por exemplo.

 Temos, portanto, um item errado ao afirmar que um dos elementos do SOAP é um protocolo de transporte.

Fonte: tecconcursos

**89** [**Q996805**](https://www.qconcursos.com/questoes-de-concursos/questoes/211431f9-86) [WebServices](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/webservices)  **Prova:**[CESPE - 2019 - SLU-DF - Analista de Gestão de Resíduos Sólidos - Informática](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2019-slu-df-analista-de-gestao-de-residuos-solidos-informatica)

Acerca de REST e DHCP, julgue item que se segue.

Entre os princípios orientadores a serem seguidos na implantação de uma API RESTful Java inclui-se o *stateless*, em que cada solicitação do cliente para o servidor deve conter todas as informações necessárias, independentemente das informações armazenadas no servidor.

**Rest**utiliza basicamente HTTP que é stateless. O servidor não precisa lembrar o estado do cliente. Os clientes devem incluir todas as informações necessárias na requisição para que o servidor possa entendê-la e processá-la.

Each request from the client to server must **contain all of the information necessary to understand the request**, and cannot take advantage of any stored context on the server.

<https://restfulapi.net/statelessness/>

Concordo, 54321, apesar de vc nao ter citado nenhum fonte, fui atras e achei algumas, como[1]. Mas isso nao significa q a questao precise ser anulada por isso. Muito pelo em ovo dessa galera!

Atencao, galera, nao vamos perder tempo com baboseiras, futilidades, inutilidades, mimimi e coisas do genero!

Vamos 'a resolucao?

Segundo[1], "*RESTful systems are meant to be****stateless****. Remember that REST defines the server stateless, which means that storing the user data in session after the initial login is not a good idea*".

Ja para[2], "*The fourth RESTful principle I will discuss is the idea of****statelessness****. When I talk about statelessness, though, I don't mean that your applications can't have state. In REST,****stateless****means that there is no client session data stored on the server. The server only records and manages the state of the resources it exposes. If there needs to be session-specific data, it should be held and maintained by the client and transferred to the server with each request as needed. A service layer that does not have to maintain client sessions is a lot easier to scale, as it has to do a lot fewer expensive replications in a clustered environment. It's a lot easier to scale up, because all you have to do is add machines*".

Fonte:

[1] Pro REST API Development with Node.js, Fernando Doglio

[2] RESTful Java with JAX-RS, Bill Burke

O correto seria "statelessness"

**Correto. Informação completa e independente.**

**90** [**Q996786**](https://www.qconcursos.com/questoes-de-concursos/questoes/20bcc4df-86) [WebServices](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/webservices)  **Prova:**[CESPE - 2019 - SLU-DF - Analista de Gestão de Resíduos Sólidos - Informática](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2019-slu-df-analista-de-gestao-de-residuos-solidos-informatica)

Julgue o próximo item, a respeito de *domain-driven design*, *design pattern*s, *emergent design*, *enterprise content management* e REST.

Em um *web service* REST que gerencie alguns tipos de serviço, os conflitos decorrentes de recursos que tenham identificadores iguais são automaticamente resolvidos no *web service*.

**Errada**

Toda aplicação gerencia algumas informações. Uma aplicação de um E-commerce, por exemplo, gerencia seus produtos, clientes, vendas, etc. Essas **coisas** que uma aplicação gerencia são chamadas de **Recursos** no modelo REST.

Um recurso nada mais é do que uma abstração sobre um determinado tipo de informação que uma aplicação gerencia, sendo que **um dos princípios do REST diz que todo recurso deve possuir uma identificação única**. Essa identificação serve para que a aplicação consiga diferenciar qual dos recursos deve ser manipulado em uma determinada solicitação.

**Gabarito: " Errado"**

O Rest, um recurso refere-se a um tipo de informação é manipulada por uma aplicação. Esses recursos devem ter identificador único, para que não ocorram conflitos. No caso do web service citado, ele não teria como saber a qual recurso se está referindo se ocorresse a situação de ter mais de um recurso com o mesmo identificador.

a ideia é que cada recurso tenha um identificador único URI

através dos diferentes métodos HTTP vc consegue fazer operaçoes diferentes

GET /pessoas/1

DELETE /pessoas/1

PUT pessoas/1

Exato. Se O servidor não possui estado, nao tem como ele saber se o ID ta duplicado.

Segundo[2]"*In REST,****stateless****means that there is no client session data stored on the server. The server only records and manages the state of the resources it exposes. If there needs to be session-specific data, it should be held and maintained by the client and transferred to the server with each request as needed".*

**Fonte:**

*[1] Pro REST API Development with Node.js, Fernando Doglio*

*[2] RESTful Java with JAX-RS, Bill Burke*

O servidor não possui estado

**91** [**Q996783**](https://www.qconcursos.com/questoes-de-concursos/questoes/20ae30aa-86) [Padrões de projeto (Design Patterns)](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/padroes-de-projeto-design-patterns)  **Prova:**[CESPE - 2019 - SLU-DF - Analista de Gestão de Resíduos Sólidos - Informática](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2019-slu-df-analista-de-gestao-de-residuos-solidos-informatica)

Julgue o próximo item, a respeito de *domain-driven design*, *design pattern*s, *emergent design*, *enterprise content management* e REST.

O uso de *design patterns* leva à unificação dos códigos utilizados em diferentes aplicações que utilizem o mesmo padrão.

Capiciosa demais essa pergunta. De fato não unifica o CÓDIGO (por isso questão errada), mas unifica os conceitos. Seria muito fácil o próprio examinador ter se confundido e colocado o gabarito como certo. Cuidado.

marcaria certa com certeza

Não necessariamente leva à unificação dos códigos e sim a sua **padronização**.

**ERRADO**

Os padrões de projeto padronizam o pensamento utilizado para resolver um problema. Cada padrão representa uma ideia de resolução de um problema comum a vários programadores, porém ele não dita o modo como essa ideia de ser implementada via código. Assim, não tem como garantir uma unificação de código, até porque os padrões de projetos independem da linguagem de programação, desde que seja um projeto OO.

Segue lá para mais dicas: @rodolfodalves

**GABARITO ERRADO**

Nada obrigada dois sistemas que utilizam o mesmo padrão de projeto serem compatíveis, porém facilita a sua integração.

Adendo: No mundo acadêmico essa questão é altamente polêmica.

**92** [**Q996781**](https://www.qconcursos.com/questoes-de-concursos/questoes/20a58537-86) [WebServices](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/webservices)  **Prova:**[CESPE - 2019 - SLU-DF - Analista de Gestão de Resíduos Sólidos - Informática](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2019-slu-df-analista-de-gestao-de-residuos-solidos-informatica)

Acerca de arquitetura de *software*, julgue o item a seguir.

Um *web service* pode assumir o papel de provedor de serviço e de consumidor de serviço.

**PROVEDOR DE SERVIÇOS**

Agindo como um provedor de serviços, um web service expõe uma interface pública através da qual pode ser chamado por solicitantes do serviço. Um provedor de serviços disponibiliza esta interface publicando uma descrição do serviço. Num modelo cliente-servidor, o provedor de serviço pode ser comparado ao servidor.

O termo “provedor de serviço” pode também ser usado para descrever a organização ou ambiente que hospeda (provê) o web service.

**Um provedor de serviço pode também agir como um solicitante de serviço**. Por exemplo, um web service pode atuar como um **provedor de serviço quando um solicitante de serviço lhe pede para executar uma função**. Pode então atuar como um**solicitante de serviço quando mais tarde contata o solicitante de serviço original (agora agindo como um provedor de serviço) para solicitar informação de status.**

Fonte: https://www.devmedia.com.br/web-services/2873

**Gabarito: " Certo"**

Web services são usados para implementar e disponibilizar serviços pela Web. Há situações em que um web service assume um papel intermediário. Nesse caso, ele pode agir como provedor de serviço , quando recebe mensagem, e também como solicitante ou consumidor de serviço, quando passa adiante a mensagem recebida.

**93** [**Q996779**](https://www.qconcursos.com/questoes-de-concursos/questoes/209d14b1-86) [Arquitetura de Software](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/arquitetura-de-software)  **Prova:**[CESPE - 2019 - SLU-DF - Analista de Gestão de Resíduos Sólidos - Informática](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2019-slu-df-analista-de-gestao-de-residuos-solidos-informatica)

Acerca de arquitetura de *software*, julgue o item a seguir.

Uma das vantagens da utilização de uma arquitetura de microsserviços é a possibilidade de isolamento de eventuais falhas no *software*.

Os **microsserviços** são uma arquitetura e uma abordagem para escrever software. Com eles, as aplicações são desmembradas em componentes mínimos e independentes. Diferentemente da abordagem **tradicional monolítica** em que toda a aplicação é criada como um único bloco, os microsserviços são componentes separados que trabalham juntos para realizar as mesmas tarefas. Cada um dos componentes ou processos é um microsserviço. É possivel isolar a falha somente daquele microserviço não afentando toda a aplicação, no momento da correção do erro da Aplicação.

cespe melhorou bastante sua definição de microsserviços

2018

Em uma arquitetura de microsserviço, caso sejam criados vários sistemas, a falha de algum deles não afetará os demais sistemas.

certa

mas é zuada essa de 2018, pq inevitalmente eles vao se comunicar, então num contexto afeta

mas essa de 2019 está bem melhor

**94** [**Q996777**](https://www.qconcursos.com/questoes-de-concursos/questoes/20921526-86) [Arquitetura de Software](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/arquitetura-de-software)  **Prova:**[CESPE - 2019 - SLU-DF - Analista de Gestão de Resíduos Sólidos - Informática](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2019-slu-df-analista-de-gestao-de-residuos-solidos-informatica)

Acerca de arquitetura de *software*, julgue o item a seguir.

O desenvolvimento de uma aplicação *web* pressupõe uma arquitetura que se baseie no uso do HTTP em conjunto com HTML.

@jcarlaop p

não necessariamente o desenvolvimento de uma aplicação *web pressupõe a utilização de html, pode ser php, java script, java, .net, etc*

JUSTIFICATIVA CEBRASPE -**ERRADO.** Existem diferentes possibilidades de arquitetura para desenvolvimento de aplicações web. A arquitetura que se baseia em HTTP foi praticamente uma regra por muito tempo. Entretanto, atualmente isso não pode mais ser considerado verdade, já que existem outros tipos de arquitetura que não dependem desse protocolo, como é o caso das arquiteturas baseadas em objetos distribuídos.

agora relendo a questão, entendi o erro. O problema não é o protocolo HTTP, e sim seu uso **JUNTO** ao HTML. Uma vez que por exemplo um serviço REST, pode retornar os dados usando HTTP(s) e JSON ao invés de HTML.

**95** [**Q990072**](https://www.qconcursos.com/questoes-de-concursos/questoes/cc0a4fc3-75) [WebServices](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/webservices)  **Prova:**[CESPE - 2019 - CGE - CE - Auditor de Controle Interno - Tecnologia da Informação](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2019-cge-ce-auditor-de-controle-interno-tecnologia-da-informacao)

No modelo REST de desenvolvimento de *web services*, o método HTTP é utilizado para determinar a operação ou o método a ser executado. A esse respeito, assinale a opção correta.

**A** O referido modelo impõe uma padronização no formato da URL a ser utilizada.

**B** O modelo REST deixa a cargo de quem provê o serviço a definição da URL a ser utilizada.

**C** No modelo em questão, os parâmetros devem ser obrigatoriamente do tipo JSon.

**D** Os parâmetros do modelo definem a requisição dos métodos a serem acessados, tornando o acesso automatizado.

**E** O método TRACE do HTTP não pode ser manipulado pelo modelo REST.

Segue abaixo os principios para a formatação da URL

**Identificação dos recursos**

A identificação do recurso deve ser feita utilizando-se o conceito de**URI (Uniform Resource Identifier)**, que é um dos padrões utilizados pela Web. Alguns exemplos de URI’s:

\* http://servicorest.com.br/produtos;

\* http://servicorest.com.br/clientes;

**Utilize URI’s legíveis**

Ao definir uma URI, utilize nomes legíveis por humanos, que sejam de fácil dedução e que estejam relacionados com o domínio da aplicação. Exemplos:

\* http://servicorest.com.br/produto (Singular);

\* http://servicorest.com.br/clientes (Plural);

**Evite adicionar na URI a operação a ser realizada no recurso**

Os recursos que uma aplicação gerencia podem ser manipulados de diversas maneiras, sendo para isso disponibilizada algumas operações para manipula-los, tais como: **criar, listar, excluir, atualizar, etc.**Exemplos:

\* http://servicorest.com.br/produtos/cadastrar;

\* http://servicorest.com.br/clientes/10/excluir;

**Evite adicionar na URI o formato desejado da representação do recurso**

É comum que um serviço REST suporte múltiplos formatos para representar seus recursos, tais como XML, JSON e HTML. A informação sobre qual o formato desejado por um cliente ao consultar um serviço REST deve ser feita via Content Negotiation.

**Fonte:** <https://blog.caelum.com.br/rest-principios-e-boas-praticas/>

Sobre a letra A, não confundam a padronização da URL com o método HTTP a ser utilizado. GET, POST, DELETE, PUT são métodos HTTP, que definem a operação a ser realizada no serviço. A URL, por sua vez, define qual será o recurso (serviço) que será utilizado, sob o qual será feita a operação definida pelo método HTTP utilizado.

Achei estranho, assinalei a A, o método REST, por se utilizar de uma URL/URI, não impõe de certa forma uma forma de acesso Restrita às características de uma URL/URI?

REST significa "Representational state transfer" e é um estilo de arquitetura para comunicação dinâmica usando HTTP e HTTPS.

Da wikipedia:

O estilo de arquitetura REST é projetado para aplicativos baseados em rede, especificamente aplicativos cliente-servidor. Mas mais do que isso, ele é projetado para uso em escala da Internet, portanto, o acoplamento entre o agente do usuário (cliente) e o servidor de origem deve ser o mais leve (loose) possível para facilitar a adoção em larga escala. Isso é alcançado criando uma camada de abstração no servidor, definindo recursos que encapsulam entidades (por exemplo, arquivos) no servidor e ocultando os detalhes de implementação subjacentes (servidor de arquivos, banco de dados, etc.). Essa abordagem permite a maior interoperabilidade entre clientes e servidores em um ambiente de escala de Internet de longa duração que cruza os limites organizacionais (confiança).

Os clientes só podem acessar recursos usando URIs. Uma representação de um recurso é outro conceito importante em REST; para garantir que as respostas possam ser interpretadas pelo maior número possível de aplicativos cliente, uma representação do recurso é enviada em formato de hipertexto. Assim, um recurso é manipulado por meio de representações de hipertexto transferidas em mensagens entre os clientes e os servidores.

A forte dissociação de cliente e servidor, juntamente com a transferência de informações baseada em texto usando um protocolo de endereçamento uniforme, forneceu a base para atender aos requisitos da Web: robustez (escalabilidade anárquica), implantação independente de componentes, transferência de dados de grande granularidade e uma barreira de entrada baixa para leitores de conteúdo, autores e desenvolvedores de conteúdo.

Na internet, encontrei este resumo que pode ajudar bastante: <https://www.brunobrito.net.br/api-restful-boas-praticas/>

**A O referido modelo impõe uma padronização no formato da URL a ser utilizada.**

**ERRADO**

Há bastante liberdade no uso do modelo

**B O modelo REST deixa a cargo de quem provê o serviço a definição da URL a ser utilizada.**

**CORRETO**

Conforme já comentado pelos colegas.

**C No modelo em questão, os parâmetros devem ser obrigatoriamente do tipo JSon*.***

**ERRADO**

Há vários padrões aceitos, como SOAP e XML.

**D Os parâmetros do modelo definem a requisição dos métodos a serem acessados, tornando o acesso automatizado.**

**ERRADO**

Tem APIs próprias para automatização de ações específicas.

**E O método TRACE do HTTP não pode ser manipulado pelo modelo REST.**

**ERRADO**

O método HTTP TRACE executa um teste de loopback de mensagem ao longo do caminho para o recurso de destino, fornecendo um mecanismo de depuração útil.

**96** [**Q950969**](https://www.qconcursos.com/questoes-de-concursos/questoes/22c57d78-fa) [Padrões de projeto (Design Patterns)](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/padroes-de-projeto-design-patterns) **Prova:**[CESPE - 2018 - BNB - Especialista Técnico - Analista de Sistema](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2018-bnb-especialista-tecnico-analista-de-sistema)

Considerando os conceitos de análise e projeto orientados a objetos, julgue o item subsecutivo.

De acordo com os padrões GRASP, a função do *low coupling* é garantir que o acoplamento entre classes ou entidades permaneça fraco, de forma a permitir a maior reutilização possível.

Os **padrões GRASP** englobam uma série de princípios baseados em conceitos de **Orientação a Objetos**. Partindo de análises que procuram definir quais as obrigações dos diferentes tipos de objetos em uma aplicação, estes *patterns*disponibilizam uma série de **recomendações**que procuram favorecer a obtenção de sistemas melhor estruturados.

Ao todo são **nove**os padrões GRASP:

· Criador (**Creator**);

· Especialista na Informação (**Information Expert**);

· Baixo Acoplamento (**Low Coupling**);

· Alta Coesão (**High Cohesion**);

· Controlador (**Controller**);

· Polimorfismo (**Polymorphism**);

· Fabricação/Invenção Pura (**Pure Fabrication**);

· Indireção (**Indirection**);

· Variações Protegidas (**Protected Variations**).

O **acoplamento**significa o quanto uma classe **depende**da outra para funcionar.

Quanto **mais baixo melhor**, isto é, quanto **menos "amarradas"** as classes menor o acoplamento.

Baixo acoplamento

O acoplamento é uma medida de quão forte um elemento está conectado, tem conhecimento ou depende de outros elementos. **O baixo acoplamento é um padrão de avaliação que determina como atribuir responsabilidades de suporte:**

menor dependência entre as classes, mudança em uma classe com menor impacto em outras, maior potencial de reutilização.

**Padrões GRASP**

**Padrões**

– Expert – Creator – Controller – Low Coupling – High Cohesion – Polymorphism – Pure Fabrication – Indirection – Don’t Talk to Strangers

**Problema**

Como minimizar dependências e maximizar o reuso?

O acoplamento é uma medida de quão fortemente uma classe está conectada, possui conhecimento ou depende de outra classe

Com fraco acoplamento, uma classe não é dependente de muitas outras classes

Com uma classe possuindo forte acoplamento, temos os seguintes problemas:

Mudanças a uma classe relacionada força mudanças locais à classe

A classe é mais difícil de entender isoladamente

A classe é mais difícil de ser reusada, já que depende da presença de outras classes

**Solução**

Atribuir responsabilidades de forma a minimizar o acoplamento

FONTE

<http://www.dsc.ufcg.edu.br/~jacques/cursos/apoo/html/proj1/proj5.htm>

<http://www2.ic.uff.br/~leomurta/courses/2008.2/es1/aula13.pdf>

**97** [**Q950931**](https://www.qconcursos.com/questoes-de-concursos/questoes/223d7bf1-fa) [SOAP (Simple Object Access Protocol)](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/soap-simple-object-access-protocol) **Prova:**[CESPE - 2018 - BNB - Especialista Técnico - Analista de Sistema](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2018-bnb-especialista-tecnico-analista-de-sistema)

Julgue o item seguinte, a respeito de *web services*.

SOAP utiliza um sistema de mensagens SMTP sobre a camada de transporte.

**Errado.**

O SOAP, em regra, utiliza o HTTP para realizar o transporte de mensagens, mas pode utilizar tbém o SMTP, FTP, JMS...

O erro está ao mencionar a camada de transporte... qdo, na verdade, ele utiliza a **camada de aplicação!!**

Questão abaixo para reforçar!

**Ano:**2013 **Prova:**CESPE - 2013 - MPOG

SOAP utiliza o protocolo HTTP como forma de transportar dados na camada de aplicação, permite troca de mensagens baseado em XML e tem seu uso relacionado a *web services*.

Certo.

SOAP é escrito em **XML**e transportado, via de regra, por **HTTP**

**Fonte:** estratégia concursos

***SOAP utiliza um sistema de mensagens SMTP HTTP sobre a camada de transporte.*APLICAÇÃO**

protocolos de comunicação como o **SOAP** (Simple Object Access Protocol) e o REST (Representational State Transfer). O protocolo **SOAP** utiliza **XML**para***enviar mensagens*** e, geralmente, **serve-se** do protocolo **HTTP** para transportar os dados

Imagino que o erro seja por que o snmp é da camada de aplicação.

soap utiliza json também.

**98** [**Q936702**](https://www.qconcursos.com/questoes-de-concursos/questoes/7d19f4be-c3) [Padrões de projeto (Design Patterns)](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/padroes-de-projeto-design-patterns)  **Prova:**[CESPE - 2018 - MPE-PI - Analista Ministerial - Tecnologia da Informação](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2018-mpe-pi-analista-ministerial-tecnologia-da-informacao)

Julgue o item subsequente, referente a *Domain Driven Design* e a *Design Patterns*.

O padrão *chain of responsibility* desacopla o emissor do receptor, fazendo passar a solicitação por uma cadeia de receptores potenciais, ou seja, ele procura evitar o acoplamento do remetente de uma solicitação.

Chain of Responsability (Evita o Acoplamento)

Evita o acoplamento do remetente de uma solicitação ao seu destinatário, dando a mais de objeto a chance de tratar a solicitação. Encadeia os objetos receptores e passa a solicitação ao longo da cadeia até que um objeto a trate.

Utilizado, quando o objeto que trata a solicitação deve ser escolhido automaticamente.

<https://uploaddeimagens.com.br/imagens/padroes_de_projetos-jpg>

2015

O chain of responsability é comportamental e tem como objetivo

**evitar o acoplamento do remetente de uma solicitação ao seu receptor,**

de modo a permitir a mais de um objeto a oportunidade de tratar solicitação.

certa

2017

O padrão***command*** tem como definição passar uma requisição entre uma lista ou objetos encadeados para a execução de uma ação ou o acionamento de um evento em um momento posterior.

errada

c-

O padrão de comportamento e encadeamento de atendentes (chain of responsibility) evita acoplamento entre solicitantes e atendentes, permitindo que mais de um objeto tenha chance de tratar a solicitação.

**99** [**Q936701**](https://www.qconcursos.com/questoes-de-concursos/questoes/7d162a3d-c3) [Padrões de projeto (Design Patterns)](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/padroes-de-projeto-design-patterns)  **Prova:**[CESPE - 2018 - MPE-PI - Analista Ministerial - Tecnologia da Informação](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2018-mpe-pi-analista-ministerial-tecnologia-da-informacao)

Julgue o item subsequente, referente a *Domain Driven Design* e a *Design Patterns*.

O padrão *observer* separa o *display* do estado de um objeto a partir do objeto em si e, sempre que houver mudança no estado do objeto, permite atualizar e refletir a mudança em todos os *displays*.

**Observer**: define uma dependência um-para-muitos entre objetos para que quando um objeto mudar de estado, os seus dependentes sejam notificados e atualizados automaticamente.

mesmo exemplo na questao de 2019

2019 CESPE

Deseja-se implementar uma solução para o desenvolvimento de um display, o qual mostra o estado de um objeto X a partir do objeto em si, utilizando-se padrão de projeto. Displays alternativos são fornecidos, e a necessidade principal é que todos eles sejam automaticamente notificados e atualizados para refletir a mudança ocorrida quando o estado do objeto mudar.

Para resolver esse problema, é mais adequado o uso do padrão

A Observer.

B Facade.

C Adapter.

D Iterator.

E Visitor.

**100** [**Q936700**](https://www.qconcursos.com/questoes-de-concursos/questoes/7d12a0d3-c3) [Arquitetura Orientada a Objetos](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-arquitetura-de-software/arquitetura-orientada-a-objetos)  **Prova:**[CESPE - 2018 - MPE-PI - Analista Ministerial - Tecnologia da Informação](https://www.qconcursos.com/questoes-de-concursos/provas/cespe-2018-mpe-pi-analista-ministerial-tecnologia-da-informacao)

Julgue o item subsequente, referente a *Domain Driven Design* e a *Design Patterns*.

No *Domain Driven Design*, o projeto de *software* baseia sua reação em eventos externos e internos, tendo como premissa uma quantidade finita de estados que enfatizam a separação entre os modelos abstratos independentes de implementação e os específicos de implementação.

**DDD - Domain Driven Design**

    Aplicável à área de negócio específica e complexa

    Não tem como foco principal a tecnologia, mas o entendimento das regras de negócio e de como elas devem estar refletidas no código e no modelo de domínio.

    Arquitetura

        Camada de Apresentação (**User Interface**)

        Camada de Aplicação (**Application**)

        Camada de Modelo (**Domain**)

        Camada de Infraestrutura (**Infrastructure**)

    Características

        Alinhamento do código como o negócio

        Favorecer reutilização

        Mínimo de acoplamento

        Independência da Tecnologia

    Padrão

        Contexto

        Um problema

        Uma solução

    Não é arquitetura em camadas

    Úbiqua

    Anticorrupção

    Escrito por

        Eric Evans

    Projeto Orientado a Domínio