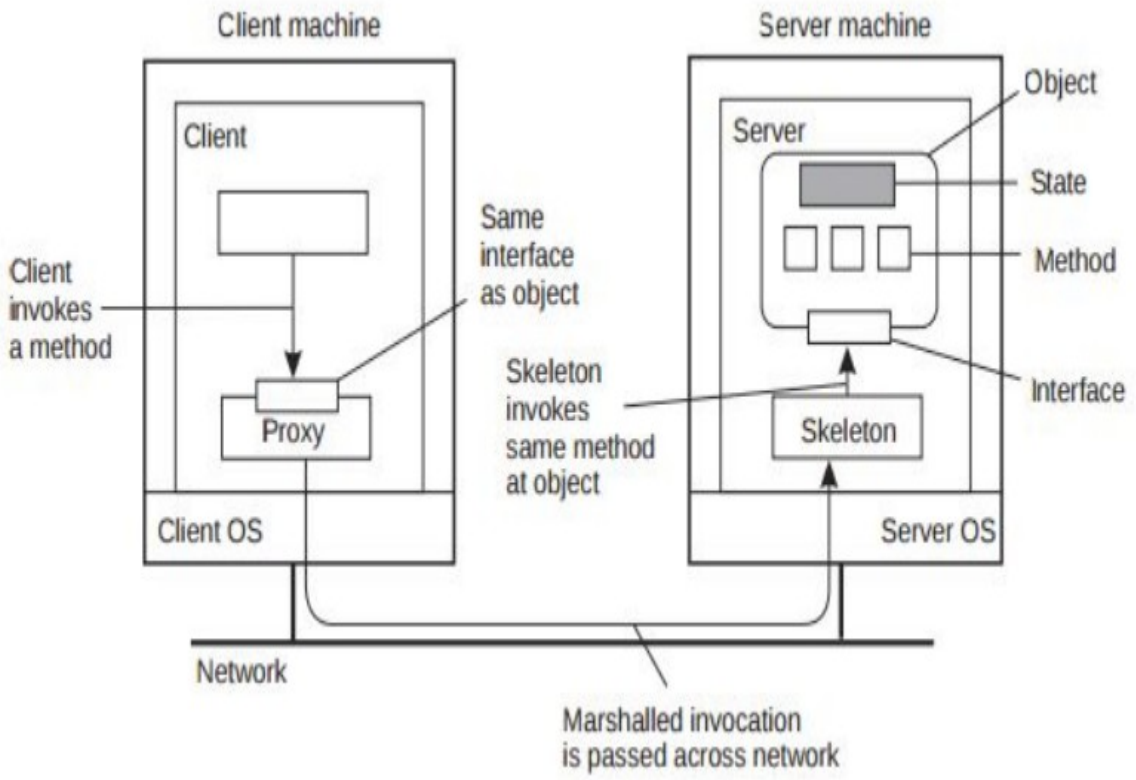
Sistemas Distribuídos Baseads em Objetos

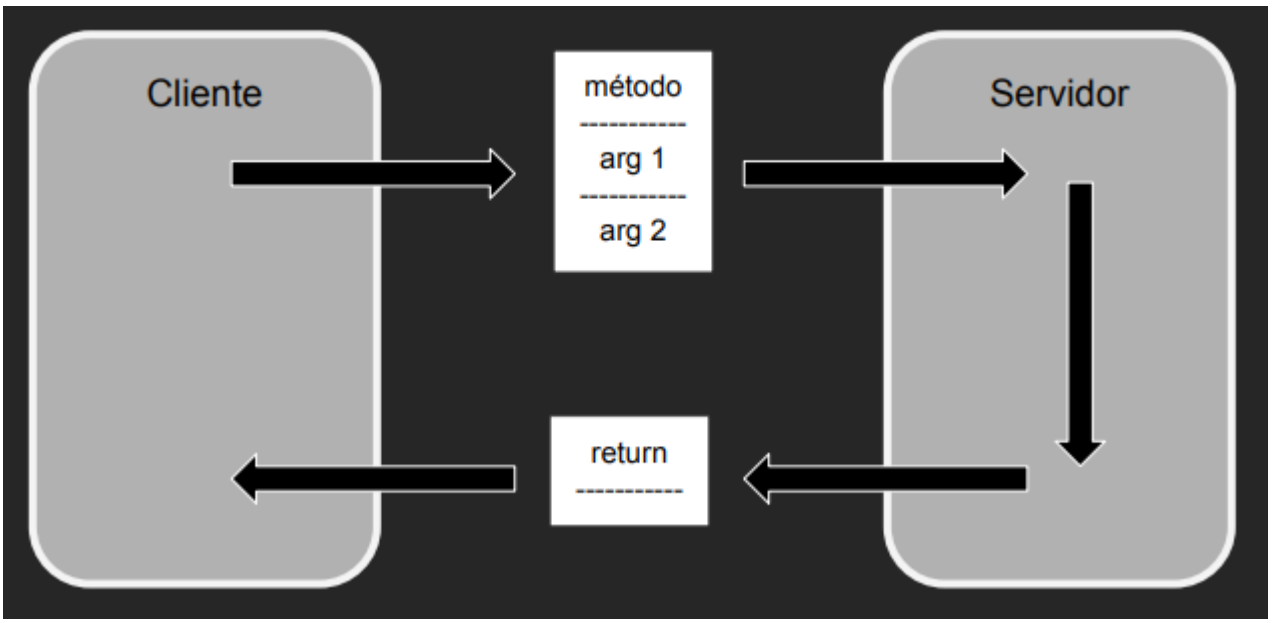
- Aplicações Distribuídas podem ser desenvolvidas empregando uma arquitetura de objetos distribuídos;
- As aplicações orientadas a objetos lidam diretamente com referências a objetos em processos remotos.
- Prover suporte à interação entre objetos, de maneira distribuída, heterogênea e transparente ao usuário.
- Os objetos não precisam ser codificados na mesma linguagem, bastando que suas interfaces sejam descritas de uma maneira padronizada.
- Um sistema de objetos distribuídos é aquele que permite a operação com objetos remotos.
- A partir de uma aplicação cliente OO é possível obter uma referência, invocar métodos desse objeto – mesmo que a instância desse objeto esteja em uma máquina diferente daquela do objeto cliente.
- O conceito básico que suporta plataformas de objetos distribuídos é o conceito de arquiteturas de objetos, que estabelece regras, diretrizes e convenções definindo como as aplicações podem se comunicar e interoperar.
- Dessa forma, o foco da arquitetura não é em como a implementação é realizada, mas sim na infraestrutura e na interface entre os componentes da arquitetura.
- A caracterísca fundamental de um objeto é que ele encapsula:
    - Dados, denominado estado, e
    - Métodos, que são disponibilizado por meio de uma interface.
- É importante entender que não há nenhum modo legal pelo qual um processo possa acessar ou manipular o estado de um objeto, exceto pela invocação dos métodos disponibilizado para ele por meio de uma interface de objeto
- Facilitar a integração de novos componentes com componentes legados
- Padrão aberto e de livre acesso
- Baseado em amplo consenso na indústria
- Localizados em diferentes dispositivos
- Comunicação via rede
- Referenciação complexa
- Troca de mensagens lenta
- Podem executar operações em paralelo
- Suscetível a ataques

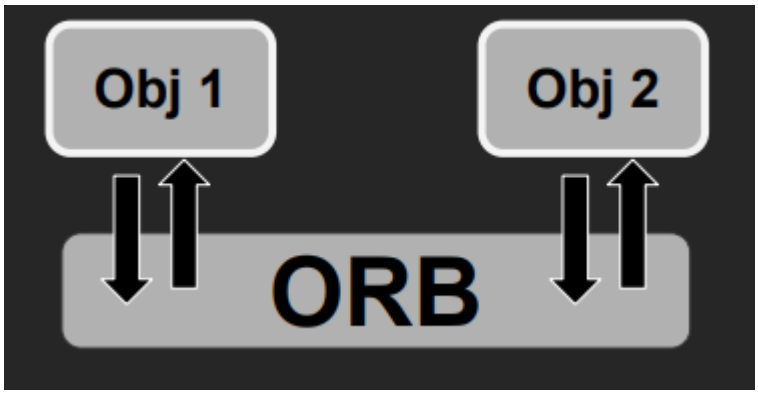


Middleware

O termo middleware se aplica a uma camada de software que fornece abstração de programação, assim como o mascaramento de heterogeneidade das redes, do hardware, de sistemas operacionais e linguagens de programação subjacentes.

- RPC - Chamada Remota de Procedimento (Remote Procedure Call)
    - Consiste em um protocolo para chamadas remotas a procedimentos como se fosse locais
    - Torna mais fácil a implementação de aplicações distribuídas, pois abstrai o código referente à parte de comunicação.
    - Serviço muito utilizado em sistemas operacionais distribuída para chamada remoto de procedimento, tais como DNS e/ou gerenciamento remoto.
    - Permite que um processo invoque um método de outro processo que esteja em seu espaço de endereço , mesmo que esteja em outro host na rede.
    - Comunicação entre cliente e servidor
    - Troca de mensagens sincronizada
    - Transparência de acesso
    - Empacota dados usando um processo de serialização
    - Usa protocolo TCP ou UDP



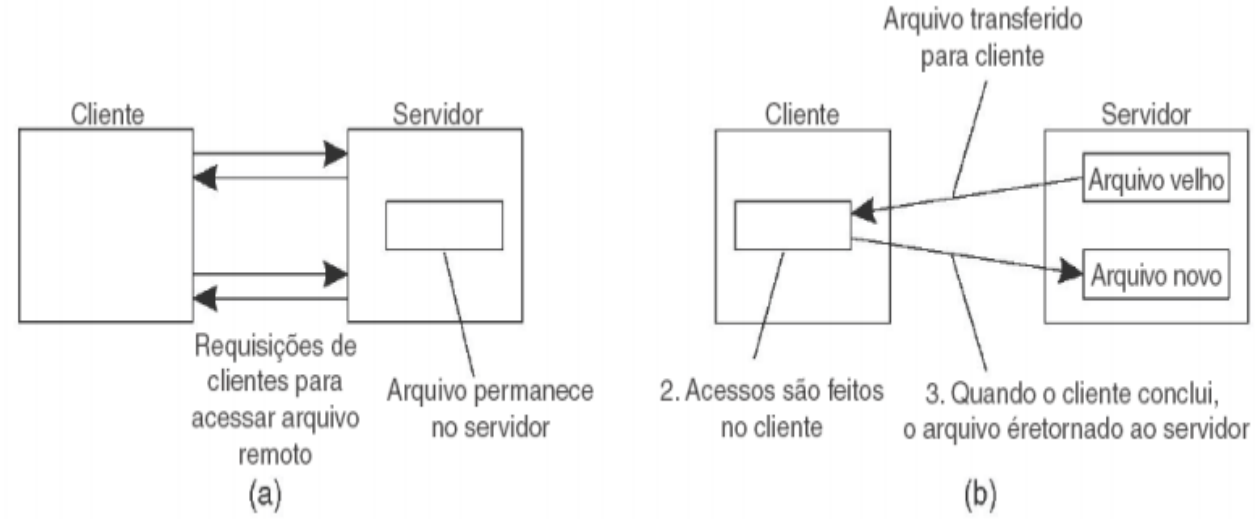
-
- ORB - Agente de Requisição de Objetos (Object Request Broker)
    - Comunicação entre objetos
    - Transparência de acesso
    - Transparência de localização
    - Usado pela arquitetura CORBA
        - é uma arquitetura para acesso a objetos distribuídos que prima pela independência da plataforma
        - A especificação CORBA defini, por exemplo, tipo de dados que podem ser mapeados para várias liguagem de programação como Java, C++ ou Pascal
    - 

- MOM - Middleware Orientado a Mensagem (Message Oriented Middleware)
- TP - Monitor de Processamento de Transações (Transaction Processing Monitor)

Sistemas de Arquivos Distribuídos

“*Um sistema de arquivos distribuídos permite aos programas armazenarem e acessarem arquivos remotos exatamente como se fossem locais, possibilitando que os usuários acessem arquivos a partir de qualquer computador em uma rede.*” (COULOURIS, et. al, p. 284).

- Foram originalmente desenvolvidos como um recurso do S.O que fornece uma interface de programação conveniente para armazenamento em disco.
- São responsáveis pela organização, armazenamento, recuperação, atribuição de nomes, compartilhamento e proteção de arquivos.
- Projetados para armazenar e gerenciar um grande número de arquivos, com recursos para criação atribuição de nomes e exclusão de arquivos.
- Permite aos programas armazenarem e acessarem arquivos remotos exatamente como se fossem locais, possibilitando que os usuários acessem arquivos a partir de qualquer computador em uma rede. O desempenho e a segurança no acesso aos arquivos armazenados em um servidor devem ser comparáveis aos arquivos armazenados em discos locais.
- A interface cliente para um sistema de arquivos é composta por um conjunto de primitivas e operações em arquivos (criar, apagar, ler, escrever)
- A interface do cliente deve ser transparente, sem distinguir entre arquivos remotos e locais Fornece um mapeamento dos nomes textuais para identificadores internos;
- Podem incluir nomes de outros diretórios.
    - Objetivo: permitir que os programas armazenem e acessem arquivos remotos exatamente como se fossem locais.
      Permitem que vários processos compartilhem dados por longos períodos, de modo seguro e confiável.
- O desempenho e segurança no acesso aos arquivos armazenados em um servidor devem ser compatíveis aos arquivos armazenados em discos locais.
- São organizados no modelo da arquitetura cliente/servidor



- Arquivos contêm dados e atributos

- Atualização concorrentes de arquivos
- Alterações feitas em um arquivo por um cliente não devem interferir nas operações de outros clientes, mesmo que esses clientes estejam manipulando o mesmo arquivo.
- Exclusão mutua.
- Serviços de arquivos atuais seguem padrões UNIX.
    - Travamentos em nível de arquivo ou registro.
    - Diferentes clientes remotos podendo acessar arquivos em servidores de arquivos.
    - Esquema de compartilhamento bem estruturado.
    - Clientes dispersos
    - Ponto de vista centralizado

Raid

Cap3 - http://bdm.unb.br/bitstream/10483/16024/1/2016_DiogoAssisFerreira_GetulioYassuykiMatayoshi_tcc.pdf

Mais do que simplesmente guardar dados, soluções de armazenamento devem fornecer acesso à informação de maneira eficiente, em tempo hábil e, dependendo do caso, oferecendo algum tipo de proteção contra falhas. É neste ponto que os sistemas RAID (Redundant Array of Independent Disks) entram em ação.

Nas próximas linhas, o InfoWester explicará o que é RAID e mostrará quais os seus principais níveis. RAID é a sigla para Redundant Array of Independent Disks ou, em tradução livre, algo como "Matriz Redundante de Discos Independentes". Trata-se, basicamente, de uma solução computacional que combina vários discos rígidos (HDs) para formar uma única unidade lógica de armazenamento de dados.

E o que é unidade lógica? Em poucas palavras, no que se refere a RAID, trata-se de fazer com que o sistema operacional enxergue o conjunto de HDs como uma única unidade de armazenamento, independente da quantidade de dispositivos que estiver em uso. Hoje, além de HDs, é possível montar sistemas RAID baseados em SSD.

Fazer com que várias unidades de armazenamento trabalhem em conjunto resulta em muitas possibilidades:

- Se um HD sofrer danos, os dados existentes nele não serão perdidos, pois podem ser replicados em outra unidade (redundância);

- É possível aumentar a capacidade de armazenamento a qualquer momento com a adição de mais HDs;

- O acesso à informação pode se tornar mais rápido, pois os dados são distribuídos a todos os discos;

- Dependendo do caso, há maior tolerância a falhas, pois o sistema não é paralisado se uma unidade parar de funcionar;

- Um sistema RAID pode ser mais barato que um dispositivo de armazenamento mais sofisticado e, ao mesmo tempo, oferecer praticamente os mesmos resultados.

Para que um sistema RAID seja criado, é necessário utilizar pelo menos dois HDs (ou SSDs). Mas não é só isso: é necessário também definir o nível de RAID do sistema. Cada nível possui características distintas justamente para atender às mais variadas necessidades. A seguir, os níveis mais comuns:

RAID 0 (zero)
Também conhecido como striping (fracionamento), o nível RAID 0 é aquele onde os dados são divididos em pequenos segmentos e distribuídos entre os discos. Trata-se de um nível que não oferece proteção contra falhas, já que nele não existe redundância. Isso significa que uma falha em qualquer um dos discos pode ocasionar perda de informações para o sistema todo, especialmente porque "pedaços" do mesmo arquivo podem ficar armazenados em discos diferentes.

O foco do RAID 0 acaba sendo o desempenho, uma vez que o sistema praticamente soma a velocidade de transmissão de dados de cada unidade. Assim, pelo menos teoricamente, quanto mais discos houver no sistema, maior é a sua taxa de transferência. Não é difícil entender o porquê: como os dados são divididos, cada parte de um arquivo é gravada em unidades diferentes ao mesmo tempo. Se este processo acontecesse apenas em um único HD, a gravação seria uma pouco mais lenta, já que teria que ser feita sequencialmente.
Por ter estas características, o RAID 0 é muito utilizado em aplicações que lidam com grandes volumes de dados e não podem apresentar lentidão, como tratamento de imagens e edição de vídeos.

RAID 1
O RAID 1 é, provavelmente, o modelo mais conhecido. Nele, uma unidade "duplica" a outra, isto é, faz uma "cópia" da primeira, razão pela qual o nível também é conhecido como mirroring (espelhamento). Com isso, se o disco principal falhar, os dados podem ser recuperados imediatamente porque existe cópias no outro.

Perceba que, por conta desta característica, sistemas RAID 1 devem funcionar em pares, de forma que uma unidade sempre tenha um "clone". Na prática, isso significa que um sistema RAID composto por dois HDs com 500 GB cada terá justamente esta capacidade, em vez de 1 TB.

O nível RAID 1 é claramente focado na proteção dos dados, ou seja, não torna o acesso mais rápido. Na verdade, pode até ocorrer uma ligeira perda de desempenho, uma vez que o processo de gravação acaba tendo que acontecer duas vezes, uma em cada unidade.

É importante observar, no entanto, que o uso de RAID 1 não dispensa soluções de backup. Como a duplicação dos dados é feita praticamente em tempo real, significa que se uma informação indevida for gravada na primeira unidade (como um vírus) ou se um arquivo importante for apagado por engano, o mesmo acontecerá no segundo disco. Por isso, RAID 1 se mostra mais adequado para proteger o sistema de falhas "físicas" das unidades.

RAID 0+1 e RAID 10

Tal como você já deve ter imaginado, o nível RAID 0+1 é um sistema "híbrido" (hybrid RAID), ou seja, que combina RAID 0 com RAID 1. Para isso, o sistema precisa ter pelo menos quatro unidades de armazenamento, duas para cada nível. Assim, tem-se uma solução RAID que considera tanto o aspecto do desempenho quanto o da redundância.

Há uma variação chamada RAID 10 (ou RAID 1+0) de funcionamento semelhante. A diferença essencial é que, no RAID 0+1, o sistema se transforma em RAID 0 em caso de falha; no RAID 1+0, o sistema assume o nível RAID 1.

RAID 5

O RAID 5 é outro nível bastante conhecido. Nele, o aspecto da redundância também é considerado, mas de maneira diferente: em vez de existir uma unidade de armazenamento inteira como réplica, os próprios discos servem de proteção. Deste modo, pode-se inclusive montar o sistema com quantidade ímpar de unidades. Mas, como isso é possível? Com o uso de um esquema de paridade.

Neste método de proteção, os dados são divididos em pequenos blocos. Cada um deles recebe um bit adicional - o bit de paridade - de acordo com a seguinte regra: se a quantidade de bits '1' do bloco for par, seu bit de paridade é '0'; se a quantidade de bits '1' for ímpar, o bit de paridade é '1'.

As informações de paridade - assim como os próprios dados - são distribuídas entre todos os discos do sistema. Via de regra, o espaço destinado à paridade é equivalente ao tamanho de um dos discos. Assim, um array formado por três HDs de 500 GB terá 1 TB para armazenamento e 500 GB para paridade.

A partir daí, se em uma tarefa de verificação o sistema constatar, por exemplo, que o bit de paridade de um bloco é '1', mas ali há uma quantidade par de bits, percebe que há um erro. Se houver apenas um bit com problema e se o sistema conseguir identificá-lo, conseguirá substituí-lo imediatamente. A restauração dos dados poderá ser feita inclusive depois de o HD ter sido trocado.

Como exemplo, imagine um bloco de dados com os bits '110X' e paridade '1'. O X indica um bit perdido, mas será que ele é '0' ou '1'? Como a paridade é '1', significa que o bloco é composto por quantidade ímpar de bits '1'. Logo, se X fosse '0', a paridade também deveria ser '0', pois ali existiria quantidade par de bits '1'. Isso significa que o bit X só pode ser '1'.

Durante a substituição, é possível manter o sistema em funcionamento, principalmente com o uso de equipamentos que suportam hot-swaping, ou seja, a troca de componentes sem necessidade de desligamento do computador. Isso é possível porque os dados são distribuídos entre todos os discos. Caso um falhe, o esquema de paridade permite recuperar os dados a partir das informações existentes nas demais unidades.

RAID 6

O RAID 5 é uma opção bastante interessante para sistemas que precisam aliar redundância com custos (relativamente) baixos, mas tem uma limitação considerável: consegue proteger o sistema se apenas um disco apresentar falha.

Uma maneira de lidar com isso é acrescentando um recurso de nome hot-spare ao sistema. Trata-se de um esquema onde um ou mais discos são acrescentados para ficar de reserva, entrando em ação tão logo uma unidade apresente problemas.

Outra alternativa interessante é o uso de RAID 6. Trata-se de uma especificação mais recente e parecida com o RAID 5, mas com uma importante diferença: trabalha com dois bits de paridade. Com isso, é possível oferecer redundância para até dois HDs no sistema, em vez de apenas um.

RAID 2, 3 e 4

Os níveis de RAID mostrados até agora são os mais utilizados, mas há alguns menos conhecidos, entre eles, RAID 2, RAID 3 e RAID 4:

RAID 2

RAID é um tipo de solução de armazenamento que surgiu no final dos anos 1980. Naquela época e nos anos seguintes, os HDs não tinham o mesmo padrão de confiabilidade que têm hoje. Por este motivo, foi criado o RAID 2. Ele é, até certo ponto, parecido com o RAID 0, mas conta com um mecanismo de detecção de falhas do tipo ECC (Error Correcting Code). Hoje, este nível quase não é mais utilizado, uma vez que praticamente todos os HDs contam com o referido recurso.

RAID 3

Este é um nível parecido com o RAID 5 por utilizar paridade. A principal diferença é que o RAID 3 reserva uma unidade de armazenamento apenas para guardar as informações de paridade, razão pela qual são necessários pelo menos três discos para montar o sistema. Este nível também pode apresentar maior complexidade de implementação pelo fato de as operações de escrita e leitura de dados considerarem todos os discos em vez de tratá-los individualmente.

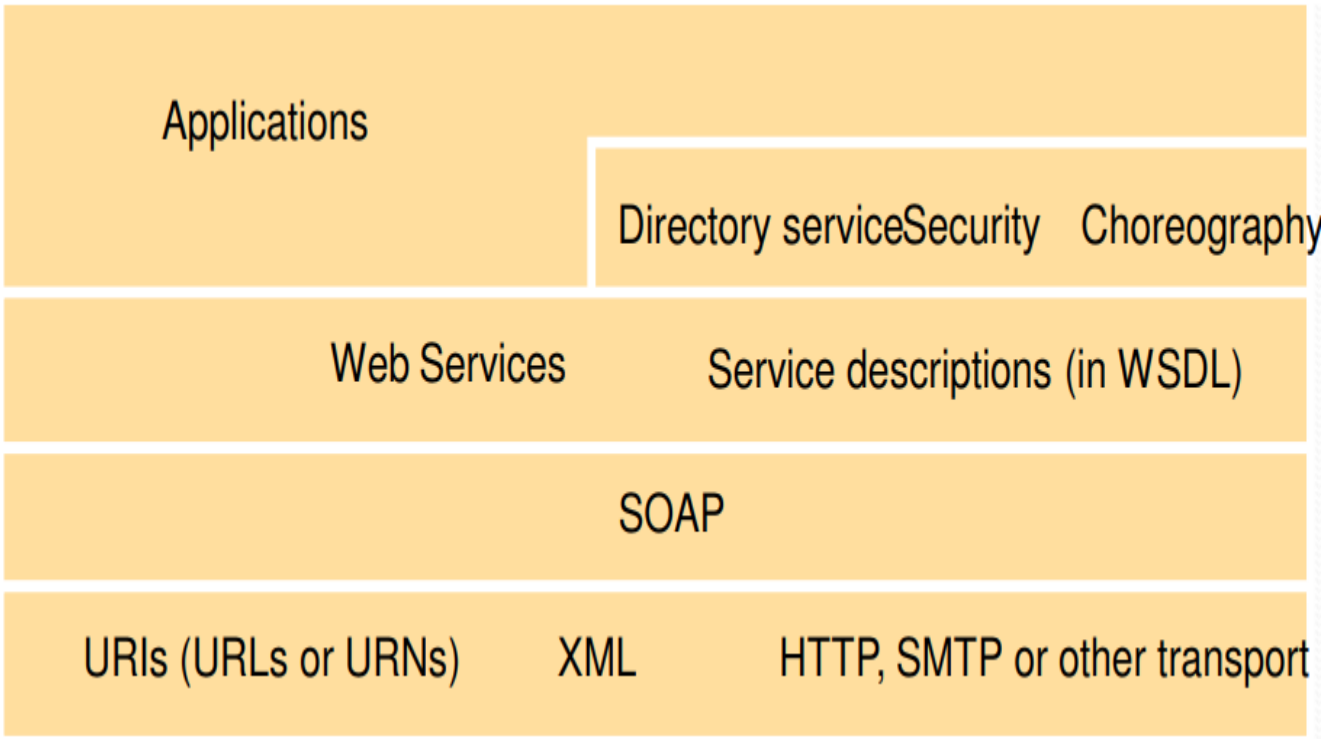
RAID 4

O RAID 4 também utiliza o esquema de paridade, tendo funcionamento similar ao RAID 3, com o diferencial de dividir os dados em blocos maiores e de oferecer acesso individual a cada disco do sistema.

Este nível pode apresentar algum comprometimento de desempenho, pois toda e qualquer operação de gravação exige atualização na unidade de paridade. Por este motivo, seu uso é mais indicado em sistemas que priorizam a leitura de dados, ou seja, que realizam muito mais consultas do que gravação.

Sistemas Distribuídos Baseados na Web

Um serviço web (web service) fornece uma interface de serviço que permite aos clientes interagirem com servidores de uma maneira mais geral do que acontece com os navegadores web.
Os clientes acessam as operações na interface de um serviço web por de meio de requisições e respostas formatadas em XML e normalmente transmitidas por HTTP. Os serviços web podem ser acessados de uma maneira ad hoc do que os serviços baseados em CORBA, permitindo que eles sejam mais facilmente usados em aplicações internet.
Assim como no CORBA e em JAVA, as interfaces dos serviços web podem ser descritas em uma IDL. Mas para os serviços web, informações adicionais precisam ser descritas, incluindo a codificação e os protocolos de comunicação em uso e o local do serviço. Os usuários exigem uma maneira segura de criar, armazenar e modificar documentos e trocá-los na internet. Os canais TLS não fornecem todos os requisitos necessários. A segurança da XML se destina a suprir essa falta.
Grade (grid) é o nome usado para referenciar uma plataforma de middleware baseada em serviços web e projetada para uso por grandes grupos dispersos de usuários, com recursos maciços de dados que exige um processamento substancial. O World-Wide Telescope é uma aplicação típica de grade para colaboração científica na área da astronomia. As características da aplicações científicas com uso intenso de dados são derivadas de um estduo do World-Wide Telescope. Essas características levaram a um conjunto de requisitos para uma arquitetura de grade.



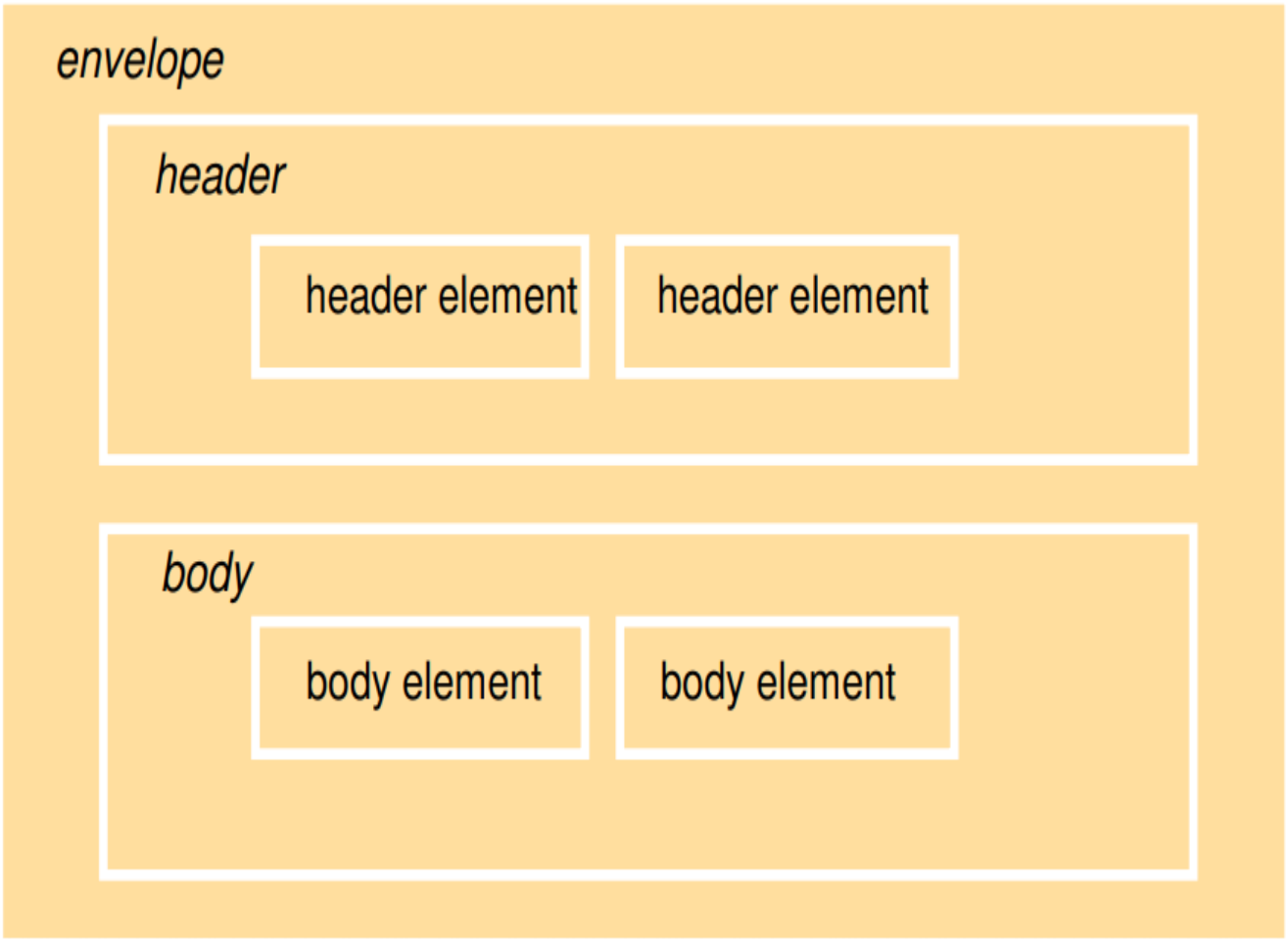
URI – uniform resource identifier – identificador de recurso geral, cujo valor pode ser um URL ou URN.
URL – inclui informações de localização do recurso, como nome de domínio do servidor de um recurso que está sendo nomeado.
URN - uniform resource names – são independentes da localização. Eles contam com um serviço de pesquisa para fazer o mapeamento para os URL dos recursos.
SOAP – especifica as regras de utilização da XML, para empacotar mensagens, por exemplo para suportar um protocolo de requisição de resposta.
XML – representação textual que embora mais volumosa do que outras representações foi adotada por sua legibilidade e pela consequente facilidade de depuração.
WSDL - Web services description language

Web Service
Geralmente uma interface de serviço web consiste em um conjunto de operações que podem ser usadas por um cliente na internet. As operações de um serviço web podem ser fornecidas por uma variedade de recursos diferentes, por exemplo programas, objetos ou bancos de dados. Um serviço web pode ser gerenciado por um servidor web, junto com páginas web, ou pode ser um serviço totalmente separado.
A principal características da maioria dos serviços web é que podem processar mensagens SOAP formatadas em XML. Uma alternativa é a estratégia REST, que está descrita em linhas gerais a seguir.
Cada serviço web usa sua própria descrição para tratar das características específicas das mensagens que recebe.
Muitos servidores web comerciais, incluindo Amazon, Yahoo, Google e eBay, oferecem interfaces de serviço que permitem ao cliente manipular seus recursos web.

O serviço web da Amazon permite aos clientes adicionar um item ao carrinho de compras ou verificar o status de uma transação. O serviços web da Amazon podem ser acessados por SOAP ou por REST.

Isso permite que aplicações de outros fornecedores construam serviços com valor agregado sobre aqueles fornecidos pela Amazon. Por exemplo, uma aplicação de controle de inventário e aquisição poderia pedir o fornecimento de mercadorias da Amazon, à medida que elas fossem necessárias e controlar automaticamente a mudança de status de cada requisição. Mais de 50.000 desenvolvedores se registraram para uso desses serviços web nos dois anos após eles serem introduzidos.
Outro exemplo interessante de aplicação que exige a presença de um serviço web é a que implementa sniping em leilões da eBay. Sniping significa fazer um lance durante os últimos segundos antes que um leilão termine.

SOAP:
O protocolo SOAP (simple object access protocol) é projetado para permitir tanto interação cliente servidor como assíncrona pela internet. Ele define um esquema para uso de XML para representar o conteúdo de mensagens de requisição e resposta, assim como um esquema para a comunicação de documentos.
Originalmente o protocolo SOAP era baseado apenas em HTTP mas a versão atual é projetada para usar uma variedade de protocolos de transporte, incluindo SMTP, TCP ou UDP.
A especificação do protocolo SOAP declara:
-Como a XML deve se usada para representar o conteúdo de mensagens individuais
- Como duas mensagens podem ser combinadas para produzir um padrão de requisição e resposta.
- As regras sobre como os destinatários das mensagens devem processar os elementos XML que elas contêm.
API’s SOAP foram implementadas em muitas linguagens: JAVA, JAVA script, Perl, Python.NET, C++, C# e Visual Basic.



Envelope: Toda mensagem SOAP deve contê- lo. É o elemento raiz do documento XML. O Envelope pode conter declarações de namespaces e também atributos adicionais como o que define o estilo de codificação (encoding style).Um "encoding style" define como os dados são representados no documento XML.
Header: É um cabeçalho opcional. Ele carrega informações adicionais, como por exemplo, se a mensagem deve ser processada por um determinado nó intermediário (É importante lembrar que, ao trafegar pela rede, a mensagem normalmente passa por diversos pontos intermediários, até alcançar o destino final). Quando utilizado, o Header deve ser o primeiro elemento do Envelope.
Body: Este elemento é obrigatório e contém o payload, ou a informação a ser transportada para o seu destino final. O elemento Body pode conter um elemento opcional Fault, usado para carregar mensagens de status e erros retornadas pelos "nós" ao processarem a **mensagem.**
Fault: contém as informações dos erros ocorridos no envio da mensagem. Apenas nas mensagens de resposta do servidor.

De acordo com o W3Schools, a estrutura da mensagem SOAP é definida em um documento XML que contém os seguintes elementos:

```
<SOAP-ENV:envelope>
    <!— Elemento raiz do SOAP e define que essa é uma mensagem SOAP-->
    <SOAP-ENV:header>
        <!—Especifica informações especificas como autenticação (opcional)-->
    </SOAP-ENV:header>
    <SOAP-ENV:body>
```

```
    <!—O elemento BODY contém o corpo da mensagem-->
  <SOAP-ENV:fault>
    <!—O elemento FAULT contém os erros que podem ocorrer-->
  </SOAP-ENV:fault>
  </SOAP-ENV:body>
</SOAP-ENV:envelope>
```

Exemplo da requisição deste envelope:

```
POST /StockQuote HTTP/1.1
Host: www.stockquoteserver.com
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
SOAPAction: "Some-URI"
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Header>
    <t:Transaction xmlns:t="some-URI" SOAP-ENV:mustUnderstand="1“>
      5
    <t:Transaction>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <m:GetLastTradePrice> xmlns:m="Some-URI">
      <symbol>DIS</symbol>
    </m:GetLastTradePrice>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Documento WSDL (Web Service Description Language)

De que forma um cliente de um Web Service sabe qual formato dos métodos a serem chamados e quais parâmetros a serem passados? Como cliente e serviço sabem como processar uma requisição? Para solucionar estes tipos de perguntas foi criado um documento, que utiliza uma linguagem chamada WSDL.

WSDL ou Web Service Description Language é uma linguagem baseada em XML, utilizada para descrever um Web Service. Um Web Service deve, portanto, definir todas as suas interfaces, operações, esquemas de codificação, entre outros neste documento.

REST (representational state transfer)

É uma estratégia com um estilo de operação muito restrito, no qual os clientes usam URL’s e as operações HTTP GET, PUT, DELETE e POST para manipular recursos representados em XML. A análise está na manipulação de recursos de dados em vez de interfaces. Os clientes recebem o estado inteiro de um recurso, em vez de chamar uma operação para fornecer alguma parte dele. Fielding acredita que, no contexto da internet, a proliferação de diferentes interfaces de serviço não será tão útil quanto um conjunto mínimo de operações simples e uniformes.

Quando um novo recurso é criado, ele recebe um novo URL por meio do qual pode ser acessado ou atualizado.

As definições de interface são necessárias para permitir que os clientes se comuniquem com os serviços. Para serviços web, as definições de interface são fornecidas como parte de uma descrição de serviço mais geral, que especifica duas outras características adicionais, como as mensagens devem ser comunicadas (por exemplo, por SOAP com HTTP) e o URI do serviço. Para fornecer serviço em ambiente com múltiplas linguagens, as descrições de serviço são escritas em XML.

A descrição do serviço forma a base de um acordo entre um cliente e um servidor quanto ao serviço oferecido. Ela reúne todos os fatos pertinentes ao serviço que são relevantes para seu clientes. As descrições de serviço geralmente são usadas para gerar stubs de cliente que implementam automaticamente o comportamento correto para o cliente.

O componente do tipo IDL de uma descrição de serviço é mais flexível do que as outras IDL's, pois um serviço pode ser especificado em termos dos tipos de mensagens que enviará e receberá, ou em termos das operações que suporta, para permitir a troca de documentos e interações estilo requisição e resposta.