

ADDETC – Área Departamental de Engenharia Eletrónica e Telecomunicações e de Computadores

LEIM -Licenciatura Engenharia informática e multimédia

**Infraestruturas Computacionais Distribuídas**

**Trabalho prático 2**

**Turma:**

LEIM-42D

**Trabalho realizado por:**

Miguel Távora Nº 45102

Pedro Henriques Nº 45415

**Docente:**

Porfírio Filipe

22/08/2020

[Introdução 2](#_Toc48747550)

[Desenvolvimento 3](#_Toc48747551)

[Armazenamento de dados no servidor 3](#_Toc48747552)

[**Mecanismos de processamento XML** 3](#_Toc48747553)

[**XML Schema** 4](#_Toc48747554)

[4. Sockets 4](#_Toc48747555)

[5. Camada de transporte TCP 5](#_Toc48747556)

[6. Seriação objetos em Java 5](#_Toc48747557)

[7. Utilização Javascript para obter páginas dinâmicas 5](#_Toc48747558)

[8. Utilização JSP para gerar páginas dinâmicas 5](#_Toc48747559)

[3.Arquitetura Desenvolvida 7](#_Toc48747560)

[1. Mensagens XML e XSD utilizados 8](#_Toc48747561)

[2. Funcionalidades extra 8](#_Toc48747562)

[3. Esquema das páginas web 9](#_Toc48747563)

[4.Conclusão 10](#_Toc48747564)

# Introdução

O segundo trabalho prático de infraestruturas computacionais distribuídas possui o objetivo de desenvolver competências sobre os conceitos associados à interação entre sistemas computacionais na *World Wide Web*.

Neste trabalho iremos explicar o processo de desenvolvimento de uma aplicação Web capaz de transmitir e receber informações desde um utilizador até um servidor, através da internet. Como a interação entre estes sistemas poderia ser feita através de diferentes conceções, iremos abordar somente o desenho implementado no trabalho prático.

Iremos explicar como é feita a comunicação entre clientes e servidor, a troca de mensagens entre estes Nomeadamente a comunicação realizada através de *Sockets*, em conjunto com o protocolo TCP. Alem da geração de páginas web dinâmicas capazes de interpretar e exibir as mensagens recebidas de forma visualmente apelativa ao utilizador. Nomeadamente através de *JavaServer Pages* e através de tecnologias do lado do cliente (*Javascript* e CSS).

Outro tema abordado neste relatório será o armazenamento de ficheiros numa base de dados, ainda que de uma forma bastante primitiva, e o acesso e a escrita para essa base de dados. Iremos explicar a comunicação através de documentos em formato de texto, nomeadamente em documentos XML, quais são as vantagens de utilização de XML para as comunicações e utilização de XSD para validar os dados existentes no XML.

# Desenvolvimento

## Armazenamento de dados no servidor

O armazenamento de dados no servidor, foi realizado através de documentos de texto que seguem conjuntos de regras definidos pelo consórcio W3C, designado por XML. XML, do inglês, *Extensible Markup Language*, é uma linguagem de marcas que permite definir um conjunto de regras que permite definir documentos num formato legível por máquinas e humanos. Tendo em conta a sua simplicidade é a linguagem *de facto* da internet permitindo a criação de bases de dados e a troca de informações de maneira simples entre máquinas. Foi também pela sua simplicidade que foi escolhido para servir de base de dados neste trabalho, ao oposto de bases de dados relacionais ou tabelas de dados.

### Mecanismos de processamento XML

#### Baseado em Texto

* Este tipo de processamento não necessita de instrumentos auxiliares ao seu processamento.
* Contudo esta abordagem é pouco produtiva e está severamente sujeita a possíveis erros.

#### Baseado em SAX (*Simple API for XML*)

* Analisador gera eventos no início/ fim do documento ou elemento, e é tratado como um fluxo unidirecional de caracteres.
* O modelo de programação baseado em eventos, durante a análise origina a chamada de procedimentos.

#### Baseado em DOM (Document Object Model)

* Documento é integralmente armazenado em memória.
* Modelo baseado em árvore.
* Permite leitura e escrita e posteriormente atualizar o documento XML

Neste trabalho prático foi utilizado sobretudo o mecanismo DOM para manipular documentos XML. O DOM é uma recomendação do consórcio W3C e define uma serie de interfaces capazes de manipular ficheiros XML e o seu conteúdo. No modelo DOM são definidas diversas interfaces Java, como por exemplo

***Document*­ –** representa todo o documento XML, também pode ser referido como árvore DOM.

***Element*** – Um elemento XML, corresponde diretamente ao que se encontra entre os caracteres ‘<’ e ’>’, por exemplo <elemento>, iria ter um elemento cujo nome seria “elemento”.

***Node* –** Tipo base do DOM. Representa um elemento e todos os elementos contidos dentro deste, todos os objetos que implementem a interface *Node*, possuem métodos que facilitam a sua manipulação. De notar que a interface prevê a existência de nós filhos, no entanto nem todos podem ter filhos, um exemplo disso é um nó de texto.

A criação de classes JAVA capazes de utilizar este mecanismo assenta na utilização de quatro interfaces. Onde se cria o *DocumentBuilderFactory* e se utiliza para criar um *DocumentBuilder*. O *DocumentBuilder* analisa e cria um Document e o Document será utilizado para manipular os nós da árvore.

### XML Schema

O *XML schema*, também conhecido como XML Schema Definition ou XSD, é uma linguagem baseada em XML que permite especificar a estrutura de um documento XML, assim como o DTD, no entanto possui todo um conjunto mais vasto e mais específico de possíveis regras e de estrutura dos dados. Esta linguagem permite então:

* Suporte a dados e derivação
* Suporte á integração com espaços de nomes
* Possui uma estrutura de tipos como String, tempo, data, decimal…
* Tipos, como tipo complexos e simples onde um tipo complexo é algo que possui mais do que uma característica no seu interior.

No nosso trabalho foi utilizado esta linguagem para validar o documento XML que continha as perguntas que o professor poderia enviar para os alunos. No seu interior utilizamos os tipos enumerados em cima, e também enumeradores para definir se a resposta do aluno poderia ter duas interpretações possíveis que seriam “certo” e “errado”.

Através da utilização de XSD, foi possível garantir a integridade dos documentos depois destes serem manipulados aquando da introdução de novas perguntas ou aquando da inicialização do servidor.

## Comunicação Servidor-Cliente

Uma parte fundamental deste trabalho é garantir a comunicação do utilizador de uma página web com o servidor, podendo este utilizador, aceder a diversos serviços fornecidos pelo servidor, tais como aceder a dados armazenados no servidor ou a criação e manipulação de dados. De modo a fornecer a comunicação entre o servidor e o cliente, foram utilizadas *sockets.*

### Sockets

Os Sockets são uma tecnologia que permite transmitir mensagens entre processos na mesma, ou entre máquinas distintas. Uma *socket* é uma porta para exterior, assim sendo tem um número onde envia e recebe as comunicações do exterior. Através do endereço da máquina (endereço IP) e número de porto, é possível comunicar através de *sockets*. Um Socket cliente e servidor são diferentes, o cliente estabelece ligação instantânea, enquanto o *SocketServer* espera ser conectado. Quando é estabelecida uma conexão o *SocketServer* passa também a ser *Socket*. Resumidamente um *Socket* é um mecanismo bidirecional que permite a comunicação entre processos na mesma máquina ou máquinas diferentes.

### 5. Camada de transporte TCP

O TCP é um protocolo de comunicação da camada de transporte do modelo OSI. Este protocolo garante a entrega sequencial e sem erros de todos os pacotes enviados através deste. O TCP é uma tecnologia orientada à conexão, fiável, sem fronteira mensagem, bidirecional e um processo pode trabalhar com várias conexões em simultâneo.

A tecnologia utilizada no trabalho é o TCP/IP, porque esta tecnologia garante o envio da mensagem para o destinatário baseado no seu endereço IP. Uma alternativa a este protocolo seria o protocolo UDP, no entanto, a tecnologia UDP não é fiável, assim apesar do menor tamanho dos pacotes UDP em comparação com TCP, foi escolhido o TCP de modo a garantir a chegada sequencial de todos os datagramas enviados.

Este protocolo foi utilizado em conjunto com as sockets para permitir o transporte de mensagens entre o cliente e o servidor.

### 6. Seriação objetos em Java

Em Java é possível decompor objetos numa sequência de bytes que armazenam toda a informação relativa a um objeto, desde a informação contida no seu interior, passando por tipo de objeto e até aos tipos de dados armazenados nas suas variáveis. Este objeto serializado pode ser escrito num ficheiro, e inclusivé enviado através de sockets para outros *hosts*.

Em Java todos os objetos podem serializados desde que implementem a interface Serializable, e que todos os objetos que constituem a classe também implementem esta interface. De notar, no entanto a reconstrução dos objetos é independente do Sistema Operativo, pelo que um objeto pode ser reconstruído numa máquina que não esteja a correr a JVM.

## Geração dinâmica de páginas web

### Javascript

*Javascript* (JS), é uma linguagem de programação interpretada de tipos fracos. Juntamente com HTML e CSS, é uma das principais tecnologias da *World Wide Web*, sendo interpretada por todos os principais browsers no mercado. Esta linguagem permite a manipulação de páginas web antes e depois de serem exibidas em browser. No entanto não é uma linguagem segura, ou seja, qualquer cliente consegue ver e alterar o código JS no seu browser.

No trabalho prático foi utilizado *JS* em diversas ocasiões. Foi através de *JS* que foi feito a primeira validação do lado do cliente, onde através de expressões regulares, tamanho do input dado pelo utilizador e verificação do tipo de dados a mensagem era transmitida ou não. Quando os dados não se encontram de acordo com os dados previstos o *JS* exibe uma mensagem de erro para o utilizador. No entanto como dito em cima, o JS pode ser contornado pelo que também foi implementado validações do lado do servidor, primeiro através de *servltes* no servidor web, e posteriormente validação no servido. O *JS* foi também utilizado para manipular a página de acordo com a utilização do cliente, nomeadamente alteração da parte gráfica.

### JSP

JSP é uma tecnologia que permite o desenvolvimento de páginas web geradas dinamicamente, utilizando a sintaxe da linguagem Java. Assim sendo é uma linguagem independente da plataforma podendo ser executada em máquinas com diversos sistemas operativos. Esta tecnologia permite produzir aplicações que acedam a bases de dados, manipular arquivos HTML, e recolher a informação introduzida pelo utilizador. Desta forma podemos escrever ficheiros com as marcas de HTML e através de scripts introduzidos diretamente na página alterar a sua aparência ou comportamento.

Esta tecnologia funciona do lado do server web, e ao ser compilada, não aparece na página web final, sendo apenas mostrado ao utilizador código HTML, no entanto exige um servidor web capaz de a suportar. Neste trabalho optamos por utilizar o *Tomcat*, seguindo a recomendação dos docentes.

Neste trabalho as JSP foram utilizadas para requisitar informação de um servidor, modificar a criação de páginas web, exibindo conteúdos distintos para cada utilizador e finalmente recolher a informação inserida pelo utilizador, validar e finalmente reencaminhar ao servidor.

# Arquitetura Desenvolvida

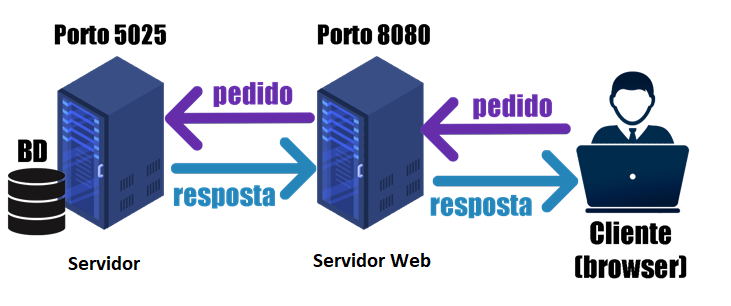


Figura 1 - Arquitetura do trabalho

A arquitetura desenvolvida possui três partes o servidor desenvolvido no trabalho prático 1, o servidor web (tomcat) e o cliente.

O servidor do trabalho 1 é o servidor responsável por armazenar as perguntas existentes numa base de dados (ficheiro XML), alem disso armazena também as credenciais de acesso dos professores, nomeadamente username e password. Este servidor recebe pedidos do servidor tomcat e retorna a respetiva resposta. Este servidor está no porto 5025.

O servidor do tomcat é o servidor responsável por gerar as páginas dinamicamente a partir da utilização do cliente. Este servidor é também responsável por toda a lógica de negócio do sistema implementado, sendo neste servidor que são guardadas as instâncias dos clientes, onde são guardadas as perguntas a serem submetidas aos alunos e as respostas dos utilizadores e posterior exibição. Este servidor por sua vez também é cliente ao servidor do tp1, pois para saber credenciais de acesso e para obter as perguntas tem de estabelecer conexão e pedir ao servidor os recursos ao que o servidor do tp1, ao qual o tp1 responde com os recursos.

O cliente é o responsável por gerar pedidos que são posteriormente enviados para o servidor do tomcat localizado no porto 8080. O cliente por sua vez é o utilizador do browser que neste trabalho assume duas variantes. O cliente pode ser um aluno ou um professor. Caso o cliente seja um professor precisa de estar registado no servidor do tp1 e é o responsável por criar a sala e enviar perguntas para os alunos. O cliente aluno é o responsável por aceder a uma sala com uma senha que foi dada pelo professor. Não existem duas salas com a mesma senha, pelo que ele entra na sala correspondente a essa senha que por sua vez corresponde a um dado professor. Quando o professor submete uma pergunta esta é recebida pelo aluno ao que este responde e a sua resposta fica guardada no servidor do tomcat.

A cada professor é lhe atribuído um id de sessão único e através de id é possível identificar os alunos conectados a este, responder adequadamente a pedidos enviados por ele. Este id é também utilizado para garantir que certas páginas só são acedidas depois de ter sido executado um login com sucesso.

## 1. Comunicação servidor – web-server

A comunicação entre o servidor web e o servidor é assegurada através do protocolo TPC/IP, assim sendo não foi preciso implementar nenhuma medida que verifique a integridade das mensagens, nem se estas chegaram ao destinatário. As portas escolhidas poderiam ser outras, no entanto foram escolhidas de modo a não gerar conflitos com outros programas em execução, mas mantendo a proximidade numérica com outras portas utilizadas para comunicação web.

Todas as mensagens transmitidas do servidor web (tomcat) para o servidor seguiam um padrão XML que permitia a sua manipulação de maneira simples, e eficiente computacionalmente através do DOM. Assim sendo o servidor e o tomcat concordam num conjunto de nós raízes que serão posteriormente interpretadas pelo servidor. Tudo desde enviar as perguntas disponíveis, para o tomcat, até introduzir novos professores surge como resposta a um pedido enviado sobre a forma de um ficheiro XML. Um bom exemplo é a criação de perguntas novas, onde a questão é introduzida pelo professor no seu browser, as informações são interpretadas pelo *web-server* através das suas *servlets*, é feita uma conversão para um objeto possível de ser serializado (*String*) enviado e posteriormente do lado do servidor, passível de ser reconstruído e interpretado.

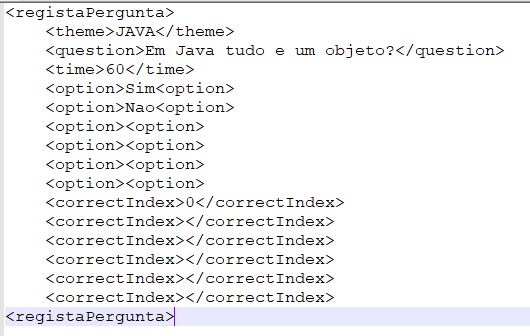
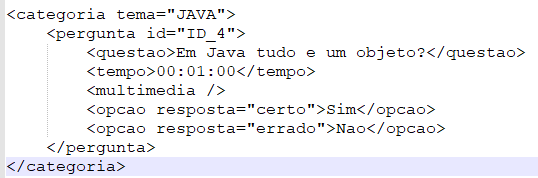
 

Figura 2 – Exemplo de mensagem e XML resultante

## Funcionalidades extra

Além das funcionalidades impostas no trabalho prático o grupo implementou três funcionalidades extra.

Uma funcionalidade extra foi a possibilidade de um cliente do tipo professor conseguir alterar a própria password na página de perfil do mesmo. Esta alteração é feita submetendo a password antiga, a password nova e a repetição da password nova. A validação é feita de tal maneira que se verifica se a password antiga corresponde a password guardada na base dados e se a password nova é igual a password de repetição. A repetição de password é para prevenir enganos do utilizador ao escrever a nova password e posteriormente não a conseguir introduzir novamente.

A funcionalidade de poder registar um professor. O registo é feito pela introdução de username que ainda não exista, de uma password e também uma credencial de acesso para prevenir que utilizador que não são professores consigam criar conta. Isto deve-se a que se for um aluno este não precisa de estar registado na base dados.

A última funcionalidade extra é a possibilidade de vários professores estarem a dar aulas a vários alunos simultaneamente sem interferir com um professor com o outro. Isto foi feito através das key’s das salas onde cada aluno quando se conecta fica atribuído a uma key de uma sala e professores com key’s de outras salas não enviam perguntas para esses alunos.

## 3. Esquema das páginas web

Em cadeiras anteriores foram abordados os conceitos de HTML e CSS pelo que neste relatório não serão aprofundados estes conceitos, no entanto durante a realização deste trabalho foi decidido manter as paginas web o mais simples possível, e apenas mostrar o mínimo possível, escondendo o restante atrás de menus que o utilizador precisaria de carregar, ou até de paginas distintas. Regra geral as páginas desenhadas e implementadas neste trabalho contêm uma imagem como fundo, um painel de conteúdo e possibilidade de um header que redireciona para a página inicial.

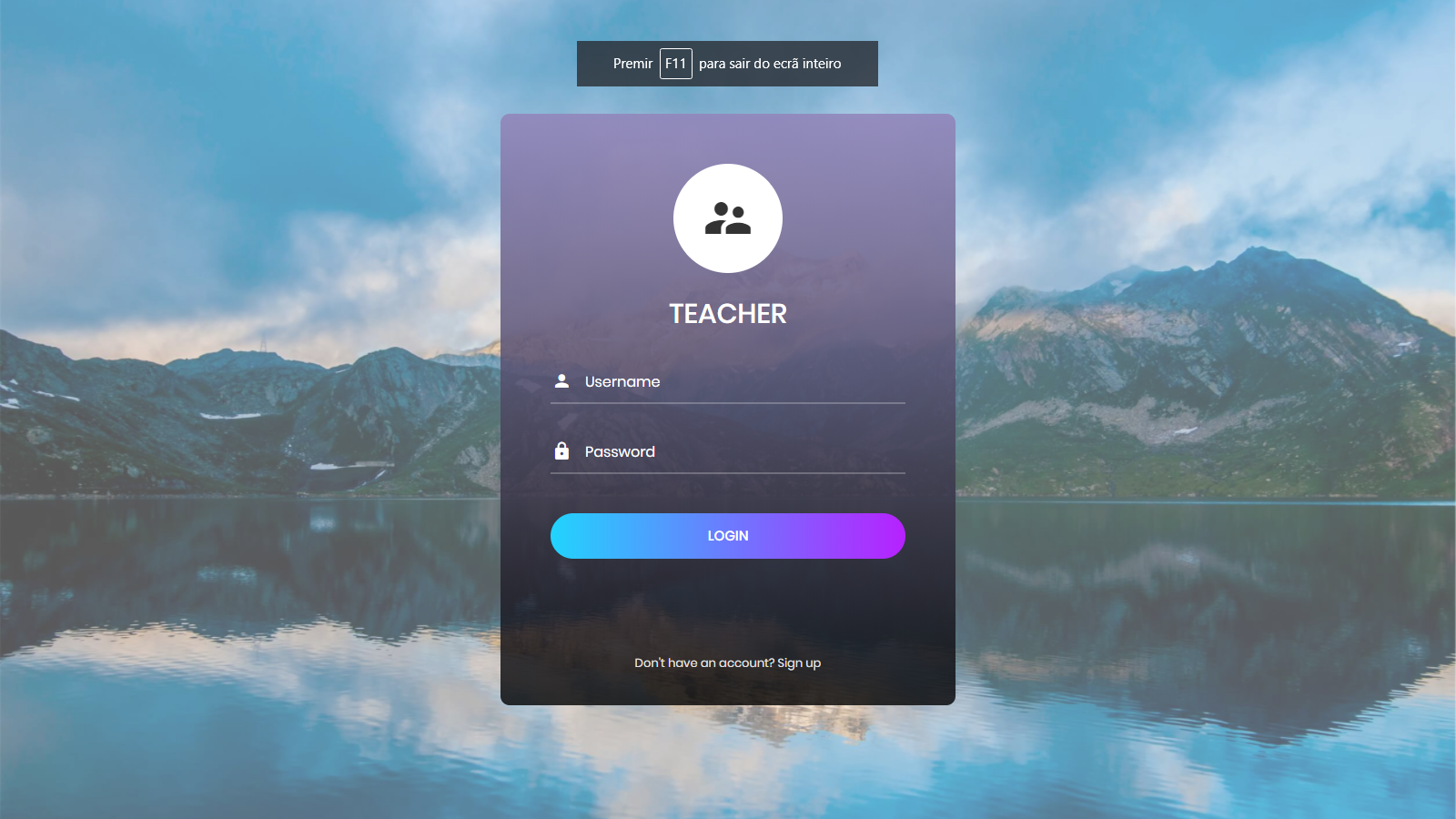


Figura 2 Página web, exemplificativa

Neste trabalho foram também introduzidas páginas web dinâmicas, desta forma foi necessário levar em consideração a quantidade de texto que poderia ser incluída, pelo que através das CSS, nomeadamente as *flex-box,* somos capazes de ajustar dinamicamente o tamanho dos elementos HTML de modo a que não ocorram artefactos estranhos na exibição dos conteúdos.

# 4.Conclusão

No presente trabalho prático o grupo cumpriu os objetivos requeridos no trabalho prático, tendo criado uma aplicação Java que funciona entre processos distintos no mesmo ou diferentes máquinas.

O grupo aprendeu ainda a construir uma arquitetura de cliente servidor e as efetivas vantagens e desvantagens do mesmo.

Observou-se de facto a utilização prática do XML em diferentes contextos, tanto para guardar em ficheiros como para enviar mensagens entre processos.