Processamento Estruturado de Informação

Trabalho Prático

Grupo 4

José Baltar, 8170212

Rodrigo Coelho, 8170282

Rui Silva, 8170283



**Índice**

[1. Introdução 3](#_Toc530841496)

[1.1. Apresentação do Problema 3](#_Toc530841497)

[1.2. Contextualização 3](#_Toc530841498)

[2. Desenvolvimento do Projeto 4](#_Toc530841499)

[2.1. Estruturação e debate 4](#_Toc530841500)

[2.2. Requisitos 4](#_Toc530841501)

[2.3. Execução e explicação 5](#_Toc530841502)

[2.4. Propriedades dos *Schemas* 6](#_Toc530841503)

[2.5. Detalhes Finais 7](#_Toc530841504)

[3. Conclusão 8](#_Toc530841505)

# Introdução

## Apresentação do Problema

Para a Unidade Curricular de **Processamento Estruturado de Informação**, foi proposto, para Avaliação Contínua, a realização de um Trabalho Prático, consistente na resposta a um Enunciado com a apresentação de um problema para sua posterior solução. Este enunciado é o seguinte:

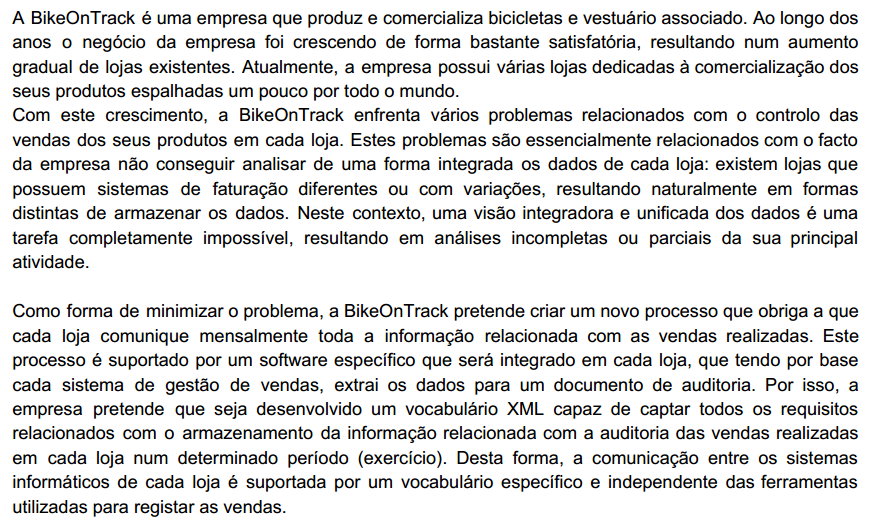


Figura 1 Enunciado do problema

## Contextualização

Para desenvolver este projeto, foi necessário utilizar conhecimento adquirido nas aulas da U.C., de forma a criar um vocabulário de fácil compreensão e interpretação, modelado de forma a que seja possível e fácil a integração de mais informação e vocabulário numa altura mais tardia, ou seja, noutras entregas futuras do trabalho.

O projeto foi, portanto, estruturado de forma a ter os *Schemas* XML com capacidade de validação de um documento XML, que irá servir de exemplo do vocabulário desenvolvido.

Este relatório servirá como apoio à interpretação do projeto, de forma a elucidar em relação à estratégia e raciocínio seguidos, segundo a interpretação do enunciado apresentado.

# Desenvolvimento do Projeto

## Estruturação e debate

Inicialmente, foi necessária a realização de um debate entre os membros do grupo face à abordagem a realizar, pois as interpretações do enunciado variavam de membro para membro. Inicialmente, foram apresentadas três possibilidades de resolução e abordagem ao problema:

* Uma loja conteria em si vários documentos. Esta abordagem pretendia aproximar-se mais de uma organização baseada nas lojas, em que estas seriam o *Root* do documento.
* Uma determinada data conteria vários documentos. Esta abordagem implicaria que cada documento estivesse “armazenado” debaixo da data da sua emissão e contendo várias lojas dentro do “exercício” de venda, e assim, portanto, a *Root* do documento seriam as datas.
* O pensamento e a interpretação seguidos trazem uma abordagem mais virada em torno do próprio Documento, que se torna a *Root*. Assim, cada documento será relativo a uma loja e especificando as vendas de um determinado “exercício”, com a sua data de emissão e loja como identificadores.

Este último raciocínio foi o considerado mais correto depois do debate, pois, como se encontra descrito no enunciado, cada loja deverá comunicar um documento, não sendo preciso criar uma estrutura para loja para esta manter o documento, ao passo que a nossa interpretação implica que a estruturação do projeto seja em torno do documento principal, isto é, o documento que irá em si conter uma identificação para a loja a qual pertence.

## Requisitos

Para tal, foi necessário estruturar o projeto consoante os “documentos” que fossem necessários, ou seja, desenvolvendo requisitos de forma a facilitar a interpretação e resolução do problema. Aqui encontra-se cada requisito que levou posteriormente ao *Namespace* e *Schema* referentes (falado em mais detalhes no próximo ponto) criados e pensados para a resolução do problema.

* ***Loja:*** Servirá para identificar a loja a que o documento e as vendas relativas pertencem. Deverá conter o nome da loja, a sua identificação fiscal, contactos e morada.
* ***Cliente:*** Servirá para armazenar os dados relativos aos clientes que estarão envolvidos nos dados das vendas. De um ponto de vista económico, isto é importante para manter algum nível de seguimento por parte da empresa/loja no cliente. Deverá conter o seu NIF e nome.
* ***Produto:*** Servirá para armazenar os dados mais específicos dos produtos que estarão identificados pelo ID nas vendas. Deste modo, como os produtos possuem informações sobre as datas que estiveram à venda, será possível identificar possíveis erros sobre o registo da venda dos produtos em datas em que estes não estejam disponíveis. Do ponto de vista económico, também é importante para uma organização perceber os melhores padrões de venda de produto, de forma a maximizar o seu potencial de lucro. O Produto deverá conter: os seus detalhes, a cor, o preço, e as datas de começo e fim de comercialização.
* ***Vendas e Receitas:*** A parte mais crítica do projeto, as “Receitas” (ou vendas) serão a parte essencial que motiva a origem do documento. Deverá identificar todas as características importantes referentes à data referida. Deverá conter: Uma Receita (contendo todos os dados referente à venda, como o subtotal das vendas, o cliente envolvido no processo a taxa de imposto e respetivas linhas de venda); a moeda utilizada, tal como a taxa de conversão quando necessário (quando a moeda utilizada é o USD, não é necessário utilizar a taxa de câmbio [isto dentro do contexto do problema]); as linhas de venda (referindo o produto envolvido, quantidade, preço por unidade e o valor total).
* ***Documento Geral:*** No *Documento* Geral, definimos a matriz completa do documento a ser gerado, definindo-o sempre com a atenção devida ao enunciado e aos documentos que foram desenvolvidos, utilizando os tipos criados anteriormente.

## Execução e explicação

Durante a execução do projeto, ou seja, durante a construção dos diversos *Schemas* que o compõem, tivemos uma abordagem que consistiu na criação dos tipos “essenciais” primeiro ao documento, criando um *Schema* geral temporário que iria servir de base para algo mais detalhado e planeado, ajudando assim na organização prévia do projeto. Com base nos requisitos, saberíamos aquilo que cada *Schema* necessitaria, deixando ao grupo o critério de como desenvolver.

Desenvolvendo em grupo e ao mesmo tempo, utilizando a plataforma *Git* para facilitar a comunicação entre os diversos desenvolvimentos do projeto, a cada membro foi atribuída uma tarefa a realizar. No entanto, e à medida que a complexidade deste aumentava, foram sendo necessárias correções e apresentadas sugestões para melhorar a dinâmica do trabalho, aumentando assim a sua qualidade global, face aos problemas que iam sendo encontrados na arquitetura original do documento. Assim, depois de uma estruturação cuidada, permitiu-se uma melhoria considerável da qualidade e da sua formatação, aumento a sua *reusabilidade* e possível integração de novos componentes noutra futura etapa de realização do projeto, além da melhoria da sua legibilidade por parte de uma pessoa terceira ao seu desenvolvimento. Acreditamos que esta foi a melhor maneira de desenvolver o projeto, aumentando a eficiência do método de construção.

Assim sendo, as etapas de forma ordenada da elaboração do projeto foram:

* Interpretação do problema
* Debate sobre a abordagem a seguir e decisão final
* Criação do Repositório GIT e estruturação do projeto
* Começo da criação dos *Schemas* e *Namespaces*
* Melhorias no projeto e modificações necessárias

(aqui temos que falar diretamente do que fizemos nos XSD, por exemplo porque é que usamos “complexType” e justificar os “minOccurs” e “maxOccurs” que tem especificado [isso eu escrevi nos comentários antes de cada elemento, é isso que se escreve aqui]). Esta parte é diferente dos requisitos porque nos requisitos só explicamos a divisão e os conceitos que fizemos nos Schemas. Aqui fica a parte técnica (depois apaga esta nota!!! Está aqui só para saberes o que escrevi.)

* Correção de erros
* Criação do XML de exemplo

## Propriedades dos *Schemas*

Referindo-nos agora diretamente às componentes mais técnicas dos *Namespace* e *Schema* criados:

* ***CommonShema ("ProjetoPEI/Grupo/Entrega1/CommonSchema"):*** Com o *CommonSchema*, definimos os nossos tipos base que irão ser continuamente utilizados por todo o documento. Assim, evita-se a definição repetida de itens permitindo que todos estes tipos sejam utilizados facilmente. Adicionalmente, a existência deste *Schema* permite uma maior *reusabilidade* de todo o processo e melhor integração de novos componentes. Este *Schema* contém principalmente *simpleTypes* que possuem restrições de uso como padrões específicos de um determinado dado ou limite de valores, etc.
* ***SchemaLoja ("ProjetoPEI/Grupo/Entrega1/Loja"):*** Aqui é definido o *complexType* referente ao elemento “Loja” existente no *Schema* principal, tal como a definição de outros tipos específicos deste contexto, por exemplo o *complexType* “typeContactos” que disponibiliza, através do indicador de ordem “choice”, a possibilidade de introduzir 1 ou mais contactos (email ou telefone), devido aos indicadores de ocurrência “minOccurs(=”1”)” e “maxOccurs=(“unbounded”)”.
* ***SchemaCliente ("ProjetoPEI/Grupo/Entrega1/Cliente"):*** *Schema* idêntico ao anterior. Aqui está definido o *complexType* referente ao elemento “Cliente” existente no *Schema* principal, contendo elementos específicos do requisito “Cliente”, tal como um atributo que se refere ao ID ou número de Cliente.
* ***SchemaProduto ("ProjetoPEI/Grupo/Entrega1/Produto"):*** Neste *Schema* está estabelecido o *complexType* com elementos relativos ao elemento “Produto” a ser utilizado no vocabulário. A particularidade deste *Schema* é que vários elementos são “opcionais”, isto é o indicador de ocorrência “minOccurs” encontra-se igualado a 0. O vocabulário está escrito desta forma uma vez que essas informações não são relevantes para o contexto do problema (a documentação de vendas) bastando a existência da identificação do produto em questão.
* ***SchemaReceitas ("ProjetoPEI/Grupo/Entrega1/Receitas"):*** *Schema* alusivo ao componente de Vendas, mais propriamente o elemento “Receitas”. Neste caso são definidos os *complexTypes* respetivos à “Moeda” e “Receita”. No primeiro é delineado um *simpleType* com uma extensão de atributo que tem como base a enumeração definida no “CommonSchema” e um atributo opcional relativo ao ID da taxa de câmbio a ser utilizada. Relativamente ao segundo, estão definidos os elementos respeitantes ao requisito “Vendas e Receitas”, sendo criado globalmente um outro *complexType* relativo às linhas de Receita pertencentes à Receita original. Cada um desses tipos possui um atributo que representa a identificação dos mesmos.
* ***SchemaDocumentoGeral ("ProjetoPEI/Grupo/Entrega1/"):*** Este é o *Schema* que representa o Documento Geral e se encontram espelhados os requisitos do problema. Aqui é onde se encontra o *Root* da instância XML e a declaração dos elementos principais, tal como as datas respetivas dos mesmos presentes em atributo de uso obrigatório.

## Detalhes Finais

Todas as modificações ao projeto encontram num repositório *Git* que poderá ser encontrado em: “<https://github.com/JoseBaltar/ProjetoPEI>”. Os *commits* realizados ao repositório procuram representar aquilo que foi sendo feito por parte dos membros do grupo, mas não são réplicas fiéis do trabalho realizado por cada um. O esforço, sobretudo na procura de erros foi geral e discutida entre os membros do grupo, embora normalmente tenha sido apenas levado para o git por um membro. A execução deste relatório igualmente recaiu na ponderação de todos os membros do grupo.

# Conclusão