**UNIVERSIDADE PAULISTA - UNIP**

Daniel Gads Melo Sousa – N948EJ-4 – CC07A30

Gabriel de Brito Silva – D0799F-3 – CC07A30

Marcelo Antônio da Silva Júnior – N743EC-6 – CC06A30

Pedro Henrique Pereira de Oliveira – N10339-6 – CC06A30

Willian de Sousa Rodrigues – D07895-0 – CC07A30

**Desenvolvimento do escopo de um projeto de um produto de software**

Implementação de um Sistema de Controle de Notas Técnicas

Bacharelado em Ciência da Computação

Turma 2019/7 – Brasília – DF

Sumário

**INTRODUÇÃO 1**

**CONCEITOS GERAIS 3**

Requisitos de Software3

Conceito3

Tipos de Requisitos e Exemplos 3

Engenharia de Requisitos4

Conceito4

Etapas 4

Modelagem Gráfica 5

UML 6

Conceito e Exemplos6

Prototipação 9

Conceitos 9

Benefícios 9

Tipos 9

**DOCUMENTO DE REQUISITOS 9**

Descrição do Problema 9

Especificação dos Requisitos de Software 7

Especificação Suplementar 11

**REFERÊNCIAS BIBLIOGRÁFICAS 12**

1. **INTRODUÇÃO**

A ONG Jovens Ambientalistas é uma instituição de vendas, constituída com a colaboração da iniciativa privada em suas ações, também é rotulado como uma das cinco maiores ONGs no ramo de vendas do Brasil.

Segundo a filosofia corporativa interna da ONG, sua missão é, “Ser uma organização de mercado, competitivo e rentável, atuando com espírito ambiental em cada uma de suas ações junto à sociedade”. A empresa possui mais de 15.000 pontos de vendas. A organização está presente em todo o país, e em mais 21 países além do Brasil.

A ONG Jovens Ambientalista tem como ramo de atividade principal no setor de vendas no cenário ambiental. O mercado da organização consiste em um ambiente para vendas e compras de produtos recicláveis, além de investimentos tecnológicos e sistemas e serviços para monitoramento ambiental.

Atualmente a ONG possui cerca de 109.000 jovens e mais 4.000 professores voluntários. A organização possui mais de 5.000 filiais, marcando presença na maioria dos municípios do país, além de filiais espalhadas em outros países.

O organograma da instituição se divide em duas camadas. A camada mais alta é a camada de gerenciamento na qual é composta por professores e pela camada de voluntários, composta por jovens ambientalistas.

A camada de gerenciamento é focada na administração dos negócios de forma geral. Cada uma das diretorias possuem uma determinada função na administração, como a controladoria, que presta contribuições importantes para a instituição e seu processo de organização, equilibrando e ajustando as dificuldades monetárias existentes na empresa.

A Diretoria de Controladoria gerencia as despesas administrativas da organização, sendo assim; todo o orçamento, todos os valores que foram “realizados”, todos os itens que possuíram gastos, e toda a sistemática na qual as despesas possuem, é de responsabilidade da própria.

O público alvo da ONG é direcionado a pessoas interessadas ao desenvolvimento sustentável, em geral são orientados e compartilhados com Pessoas Físicas e Pessoas Jurídicas que tenham necessidade de investir com cuidado com meio ambiente. Atendendo todas as classes sociais.

O ambiente institucional tem um campo vasto para aplicação da prática do marketing voltada para o mercado consumidor. Suas estratégias precisam estar relacionadas às duas esferas, atender o mercado consumidor através das necessidades do mundo atual e acompanhando as tendências globais.

A concorrência da ONG são todas as outras principais instituições com os mesmos intuitos, sendo que hoje a organização é a segunda maior do país. No ano de 2016, a sua principal concorrente lucrou R$ 21,6 bilhões – 7,4% a menos que os ganhos de 2015, mesmo assim tornou-se a maior ONG Financeira do país com 59,8 milhões de clientes, superando a organização de Jovens Ambientalistas.

O presente trabalho tem como objetivo geral descrever de que maneira a aplicação do desenvolvimento do escopo de um projeto de um produto de software auxilia um controle financeiro e de pessoal informatizado através da Implementação de um Sistema de Controle de Notas Técnicas, com a finalidade de analisar a vantagem de um melhor controle das suas finanças e dos seus recursos humanos na/o ONG Jovens Ambientalistas.

1. **CONCEITOS GERAIS**

Esta secção é caracterizada por apresentar todo o embasamento teórico utilizado para a elaboração desse documento, utilizando de obras de autores renomados em suas respectivas áreas.

* 1. **REQUISITOS DE SOFTWARE**

Aqui são explicados todos os conceitos, abordagens e exemplos sobre um Requisito de Software.

* + 1. **CONCEITO**

De acordo com o Dicionário da Língua Portuguesa, requisito é uma condição necessária para se alcançar um determinado fim; condição.

Uma outra definição foi apresentada por Leite (1994): “Requisitos: Condição necessária para a obtenção de certo objetivo, ou para preenchimento de certo objetivo. “ (LEITE, 1994).

Na perspectiva de Swebok (2004), Requisitos são condições estabelecidas que precisam ser cumpridas de modo que a atividade requerida seja realizada. Pode-se atender por requisitos uma definição documentada de uma propriedade ou comportamento de um produto que deva ser atendida para o cumprimento da mesma.

Requisitos também são usados nas áreas de Ciência da Computação, Engenharia de Software e Engenharia de Sistemas referindo-se como necessidades, atributos ou qualidades que um sistema deve prover para ser útil ao usuário.

* + 1. **TIPOS DE REQUISITOS E EXEMPLOS**

Dentre os requisitos, estão os Requisitos Funcionais (RF) e os Requisitos Não-Funcionais (RNF).

Os Requisitos Funcionais descrevem as funções que um sistema deve conter, como deve se comportar em determinadas situações e até mesmo declarar o que o sistema não deve fazer (SOMMERVILLE; SAWYER, 1997).

Já os Requisitos Não-Funcionais descrevem não o que um sistema fará, mas como deve ser executado cada função de um sistema, tendo um papel muito importante para o desenvolvimento.

Por exemplo, um RF deverá afirmar que o software tem que apresentar facilidades para autenticar senhas do usuário e RNF deverá afirmar que o tempo de autenticação tem que ser de 4 segundos ou menos (SOMMERVILLE; SAWYER, 1997).

* 1. **ENGENHARIA DE REQUISITOS**

Aqui são explicados todos os conceitos, abordagens e exemplos sobre a Engenharia de Requisitos.

* + 1. **CONCEITOS**

Segundo Pressman (2011) compreender os requisitos de um problema é considerado um dos processos mais difíceis enfrentados por um engenheiro de software. Vendo de fora a engenharia de requisitos parece até simples, mas olhando de perto não é bem assim que funciona, já que os clientes e usuários finais em muitos casos não tem um bom entendimento das funcionalidades do sistema e mesmo que soubessem se alterariam ao longo do percurso.

Sommerville (2011) afirma que a engenharia de requisitos é o processo de descobrir, analisar, documentar e verificar os requisitos do sistema. Delimita o escopo do projeto, identifica os requisitos do cliente e dos usuários, permite avaliar se o produto do desenvolvimento atenderá as necessidades do cliente e é importante na estimativa de esforço, recursos e tempo de projeto.

* + 1. **ETAPAS**

Para Pressman (2011) ela abrange sete processos distintos: concepção, levantamento, elaboração, negociação, especificação, validação e gestão. Também é possível perceber que algumas delas funcionam em paralelo e que todas são adaptadas as necessidades do projeto. Esses processos são descritos de acordo com Pressman (2011) como:

**Concepção**. Em alguns casos, começa por uma conversa informal onde é tudo que precisa para iniciar um trabalho de engenharia de software. Mas na maioria dos casos, se inicia quando é identificado uma necessidade de negócios ou é descoberto um novo serviço ou mercado potencial.

**Levantamento**: Mesmo que pareça simples, já que consiste em perguntas aos interessados sobre os objetivos do sistema, é bem difícil. Christel e Kang identificaram problemas que são encontrados ao longo do processo. Problemas de escopo onde as informações são passadas de forma precária ou desnecessária que confundem os objetivos globais do sistema, de entendimento onde os clientes/usuários não tem o entendimento necessário para passar as informações de necessidades ao engenheiro e problemas de volatilidade que os requisitos mudam com o tempo.

**Elaboração:** Fase onde as informações que foram obtidas nas fases anteriores formam um modelo de requisitos refinados que expande ainda mais todas as necessidades de um modelo.

**Negociação:** São negociadas as características importantes de cada requisito como o custo. Com essa negociação é formado um plano de projeto realista.

**Especificação:** Poder ser qualquer documento que expresse as informações e especificações de forma coerente, persistente e compreensível.

**Validação:** Todos os artefatos produzidos anteriormente passam por uma avaliação técnica para verificar se contém algum erro, inconsistência e outras características que possam prejudicar o projeto. Uma equipe de revisão que é composta por todos os interessados pelo projeto que fazem uma validação das informações.

**Gestão de requisitos:** É certo que os requisitos mudam ao longo do projeto, com isso em mente, essa atividade faz o acompanhamento das necessidades e suas mudanças.

* 1. **MODELAGEM GRÁFICA**

O processo de Modelagem Gráfica se baseia em transformar ações ou processos de em forma de imagens ou em qualquer outro tipo de informação gráfica, seguindo passos e modelos padrões para a situação. Aqui são explicados todos os conceitos, abordagens e exemplos sobre o processo de modelagem gráfica escolhida, a *UML*.

* + 1. **UML**

Nessa parte é apresentada a linguagem *UML*, bem como seus conceitos e exemplos.

* + - 1. **Conceitos e Exemplos**

Segundo Alberto Silva e Carlos Videira (2001, p. 117) "O UML (*Unified Modeling Language*) é uma linguagem para especificação, construção, visualização e documentação de artefatos de um sistema de software". Ela é desenvolvida em parceria com diversas outras empresas, mas à sua frente está a *Object Management Group* (OMG), uma organização internacional e sem fins lucrativos que atua na padronização de diversos objetos na área da computação.

No contexto de modelagem de objetos de software, a UML se mostra extremante útil, como podemos avaliar novamente nas falas dos professores Silva e Videira:

“O UML alarga o âmbito de aplicações alvo comparativamente com outros métodos existentes, designadamente porque permite, por exemplo, a modelação de sistemas concorrentes, distribuídos, para a Web, sistemas de informação geográficos, etc.” (SILVA; VIDEIRA, 2001, p.118).

E ainda:

“A ênfase do UML é na definição de uma linguagem de modelação   
standard, e por conseguinte, o UML é independente das linguagens de   
programação, das ferramentas CASE, bem como dos processos de   
desenvolvimento.” (SILVA; VIDEIRA, 2001, p.118).

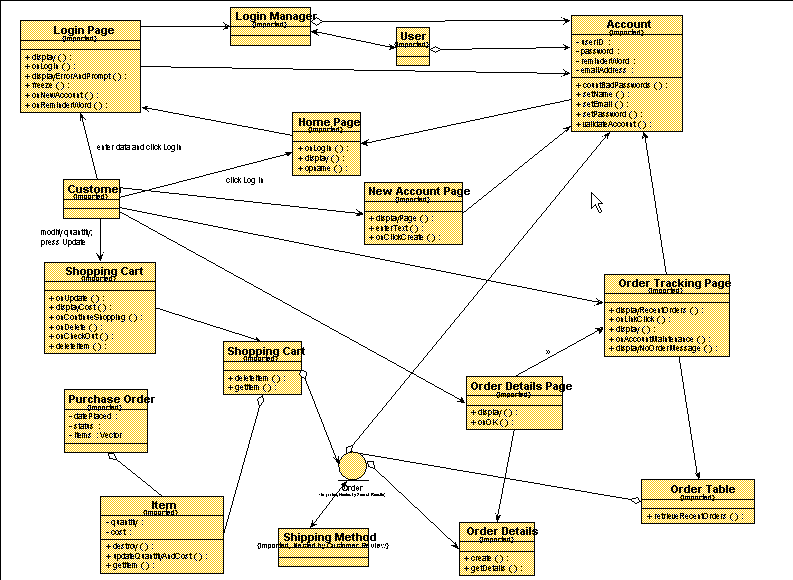
Conforme citado acima, pode-se dizer que a *Unified Modeling Language* é a linguagem padrão para ilustrar todos os componentes do software. Neste contexto, fica claro que toda a especificação e ilustração pode ser feita por meio dele. O mais importante, contudo, é que o autor deixa claro que é uma representação padrão, ou seja, independente do país em que estivermos, das linguagens de programação utilizadas ou dos métodos utilizados na empresa, poderemos usá-lo como língua única.

Com essa proposta, ela se torna a linguagem mais utilizada para tal fim, sendo assim:

“É uma linguagem de propósito geral que pode ser aplicada a todos os domínios de aplicação. Essa linguagem tornou-se, nos últimos anos, a linguagem-padrão de modelagem adotada internacionalmente pela indústria de engenharia de software. ” (GUEDES, 2011, p. 19).

Conforme citado, a *UML* é uma forma de representação de software com todas as suas dependências. Trata-se inegavelmente da maior e mais famosa linguagem de documentação da solução proposta. Seria um erro, porém, atribuir o título de linguagem de programação para a mesma. Assim, reveste-se de particular importância lembrar que ela não é uma linguagem para a programação, mas sim para a modelagem (GUEDES, 2011, p. 19). Sob essa ótica, ganha particular relevância devido a sua padronização e simplicidade.

Pode-se dizer que toda representação de software e dos seus componentes deve então seguir regras específicas e bem aplicadas para a sua utilização. Neste contexto, fica claro que para ser aplicada de forma salutar são necessárias certas abrangências, como nos assegura os professores Alberto e Carlos (2001, p. 119), por exemplo, sobre alguns elementos da *UML*: "Diagramas de casos de utilização, Diagramas de classes e diagramas de objetos, Diagramas de comportamento e Diagramas de arquitetura". Ao abordar todos os diagramas envolvidos desde a concepção até a efetiva implementação do produto, temos certeza que a representação nos auxilia e facilita na demonstração da solução.

Na figura abaixo é exemplificado um processo de *UML* para realizar *login* em uma página:

Fonte: Instituto Politécnico do Porto (2002)

Figura 1 - Exemplo de Aplicação do UML

* 1. **PROTOTIPAÇÃO**

Aqui são explicados todos os conceitos, abordagens e exemplos sobre a Prototipação.

* + 1. **CONCEITOS**

Os protótipos são utilizados como versões de teste para apresentar ao cliente uma determinada função ou produto para ser avaliada para assim entender a proposta.

* + 1. **BENEFÍCIOS**

Com o uso dos protótipos é possível ter uma avaliação direta do cliente na parte de elucidação dos requisitos para assim entender a sua necessidade.

1. **DOCUMENTO DE REQUISTOS**

Nessa seção são apresentados e discorridos os Requisitos Funcionais e Não Funcionais presentes no sistema:

* + 1. **ESPECIFICAÇÕES DOS REQUSITOS DE SOFTWARE**

Estão listados aqui os Requisitos Funcionais do Software associados as funções que o sistema deve desempenhar:

[RF001] Realizar Login.

**Ator**: Gerente, Proprietário, Atendente.

**Entradas e pré condições**: Campos para inserir informações limpos, ator entra com a sua matrícula e senha do sistema autenticador da ONG.

**Saídas e pós condições**: Se o usuário e senha estiverem corretos, o sistema prossegue, se não, uma tela de erro será apresentada.

[RF002] Manter Funcionários.

Este processo irá permitir incluir, alterar, excluir e consultar um funcionário.

**Ator**: Atendente.

**Entradas e pré condições**: Campos para inserir informações limpos, ator entra com o nome e a matrícula do funcionário.

**Saídas e pós condições**: Se as informações estiverem corretas, uma tela será apresentada para realizar as operações escolhidas, senão, uma tela de erro será apresentada.

[RF003] Manter Clientes.

Este processo irá permitir incluir, alterar, excluir e consultar um cliente.

**Ator**: Atendente.

**Entradas e pré condições**: Campos para inserir informações limpos, ator entra com o nome do cliente.

**Saídas e pós condições**: Se as informações estiverem corretas, uma tela será apresentada para realizar as operações escolhidas, senão, uma tela de erro será apresentada.

[RF004] Manter Produtos.

Este processo irá permitir incluir, alterar, excluir e consultar um produto.

**Ator**: Atendente.

**Entradas e pré condições**: Campos para inserir informações limpos, ator entra com o nome e o código do produto.

**Saídas e pós condições**: Se as informações estiverem corretas, uma tela será apresentada para realizar as operações escolhidas, senão, uma tela de erro será apresentada.

[RF005] Manter Vendas.

Este processo irá permitir incluir, alterar, excluir e consultar uma venda.

**Ator**: Atendente.

**Entradas e pré condições**: Campos para inserir informações limpos, ator entra com o nome do cliente e o produto que foi vendido.

**Saídas e pós condições**: Se as informações estiverem corretas, uma tela será apresentada para realizar as operações escolhidas e será gerado um código para a venda, senão, uma tela de erro será apresentada.

[RF005] Manter Despesas.

Este processo irá permitir incluir, alterar, excluir e consultar uma despesa.

**Ator**: Gerente, Atendente.

**Entradas e pré condições**: Campos para inserir informações limpos, ator entra com o nome da despesa e o valor gasto da mesma.

**Saídas e pós condições**: Se as informações estiverem corretas, uma tela será apresentada para realizar as operações escolhidas e será gerado um código para a despesa, senão, uma tela de erro será apresentada.

[RF006] Emitir Relatório de Despesas.

**Ator**: Gerente, Atendente.

**Entradas e pré condições**: Campos para inserir informações limpos, ator entra com o código da despesa, ou faz uma pesquisa por data.

**Saídas e pós condições**: Se as informações estiverem corretas, uma tela será apresentada com a/as despesas realizadas no dia esolhido ou pelo código informado, senão, uma tela de erro será apresentada.

[RF006] Emitir Relatório de Vendas.

**Ator**: Gerente, Atendente.

**Entradas e pré condições**: Campos para inserir informações limpos, ator entra com o código da venda, ou faz uma pesquisa por data.

**Saídas e pós condições**: Se as informações estiverem corretas, uma tela será apresentada com a/as vendas realizadas no dia esolhido ou pelo código informado, senão, uma tela de erro será apresentada.

* + 1. **ESPECIFICAÇÃO SUPLEMENTAR**

Esta seção descreve os requisitos não funcionais associados à facilidade de uso da interface com o usuário, material de treinamento e documentação do sistema.

[NF001] O sistema deverá ser implementado em Java.

[NF002] O Sistema Operacional a ser executado deverá ser *Microsoft Windows*.

[NF003] O sistema deverá usar Banco de Dados *Oracle*.

[NF004] O Hardware para ser executado deverá ter memória mínima de 2 GB de RAM e armazenamento de no mínimo 1 GB.

[NF005] O sistema deverá ocupar, no máximo, 300 MB de disco e consumir em média de 1 GB de RAM.

[NF006] O sistema deverá ter confiabilidade ligado ao tempo médio e a disponibilidade.

[NF007] O sistema deverá ter facilidade pela simplicidade do cadastro de informações.

1. **REFERÊNCIAS BIBLIOGRÁFICAS**

GUEDES, G. **UML 2 - Uma Abordagem Prática**. 2ª. ed. São Paulo: Novatec, v. I, 2011.

PRESSMAN, R. **Engenharia de Software:** Uma Abordagem Profissional. 7ª. ed. Porto Alegre: AMGH, v. I, 2011.

SILVA, A.; VIDEIRA, C. **UML Metodologias e Ferramentas CASE**. 1ª. ed. Porto: Centro Atlântico, v. I, 2001.

SOMMERVILLE, I. **Engenharia de Software**. 9ª. ed. São Paulo: Pearson Education do Brasil, v. I, 2011.