**FUNDAÇÃO ALAGOANA DE PESQUISA, EDUCAÇÃO E CULTURA**

**FACULDADE DE TECNOLOGIA DE ALAGOAS**

**ANALISE E DESNVOLVIMENTO DE SISTEMA**

**Douglas Oliveira Barbosa**

**Joalison Cabral Pontes**

**Valter Barros**

**ESPECIFICAÇÃO E DOCUMENTAÇÃO DE SOFTWARE**

### (Calculadora)

**MACEIÓ – AL**

**2015**

**FUNDAÇÃO ALAGOANA DE PESQUISA, EDUCAÇÃO E CULTURA**

**FACULDADE DE TECNOLOGIA DE ALAGOAS**

**ANALISE E DESNVOLVIMENTO DE SISTEMA**

**Douglas Oliveira Barbosa**

**Joalison Cabral Pontes**

**Valter Barros**

**ESPECIFICAÇÃO E DOCUMENTAÇÃO DE SOFTWARE**

### (Calculadora)

Trabalho elaborado para obtenção da nota parcial na disciplina de Especificação de Sistemas do Curso de Análise e Desenvolvimento de Sistemas na Faculdade de Tecnologia de Alagoas lecionado pelo prof. Victor Augusto Fragoso Florentin.

**MACEIÓ – AL**

**2015**

**SUMÁRIO DE IMAGENS**

Figura 1 - Processo XP..............................................................................................................10

Figura 2 –Diagrama de Caso de Uso........................................................................................11

Figura 3 – Levantamento de Classes.........................................................................................12

Figura 4 – Diagrama de Classes................................................................................................12

Figura 5 – Diagrama de Sequência..............................................................................................13

Figura 6 – Diagrama de Componente.......................................................................................14

Figura 7 – Diagrama de Estado...................................................................................................15

Figura 8 – Diagrama de Implantação..........................................................................................16

Figura 9 – Diagrama de Atividade..............................................................................................17

**SUMÁRIO**

1 - INTRODUÇÃO.....................................................................................................................5

2 – ENG. DE REQUISITOS.......................................................................................................6

2.1. REQUISITOS FUNCIONAIS.............................................................................................6

2.2. REQUISITOS NÃO FUNCIONAIS...................................................................................6

2.3. VIABILIDADE....................................................................................................................6

2.4. PRAZO.................................................................................................................................7

2.5. REGRAS DE NEGOCIO.....................................................................................................7

3 – PROBLEMATICA................................................................................................................8

4 - ESCOPO................................................................................................................................9

5 – CICLO DE VIDA DO SOFTWARE..................................................................................10

6 – DIAGRAMAS UML...........................................................................................................11

6.1 – DIAGRAMA DE CASO DE USO..................................................................................11

6.2 – LEVANTAMENTO DE CLASSES................................................................................11

6.3 – DIAGRAMA DE CLASSES...........................................................................................12

6.4 – DIAGRAMA DE SEQUENCIA......................................................................................13

6.5 – DIAGRAMA DE COMPONENTE.................................................................................13

6.6 – DIAGRAMA DE ESTADO.............................................................................................14

6.7 – DIAGRAMA DE IMPLANTAÇÃO...............................................................................15

6.8 – DIAGRAMA DE ATIVIDADE......................................................................................16

7 - CONCLUSÃO.....................................................................................................................18

REFERÊNCIAS........................................................................................................................19

**1 - INTRODUÇÃO**

A calculadora e um sistema que foi criado na plataforma netbeans e desenvolvida na linguagem de programação java com seu sistema voltado a ajuda o usuário a resolver cálculos com a maior rapidez e precisão, com isso melhorando seu desempenho na execução de atividades na empresa o em outros deveres.

Essa calculadora foi projetada para resolver as operações de soma, subtração, multiplicação e divisão.

A primeira máquina de calcular foi o ábaco, que surgiu na antiga mesopotâmia por volta de 3500 a.C. Os primeiros ábacos eram desenhados no chão e depois colocavam as bolas de pedras para calcular, mas com o passar do tempo, passaram a ser construídos em tabuas de pedra ou mármore, onde se lapidava as letras e se colocava depois bolas de pedras, assim fazendo os cálculos.

**2 – ENG. DE REQUISITOS**

**2.1 - REQUISITOS FUNCIONAIS**

“Os requisitos funcionais são as declarações de serviços que o sistema deve fornecer como sistema deve reagir a entradas específicas e como o sistema deve se comportar em determinadas situações. Em alguns casos, os requisitos funcionais podem também estabelecer explicitamente o que o sistema não deve fazer. “(SOMMERVILLE, 2007),

Os Requisitos funcionais aplicados na implementação da calculadora são:

O sistema precisa calcular as quatro operações fundamentais entre dois números reais.

**2.2 - REQUISITOS NÃO FUNCIONAIS**

“São restrições sobre os serviços ou as funções oferecidos pelo sistema. Eles incluem restrições de timing, restrições sobre o processo de desenvolvimento e padrões.“ (SOMMERVILLE, 2007)

Essas são os Requisitos não funcionais a respeito da calculadora:

O software precisa ser desenvolvido baseado em programação orientada a objetos.

O layout deve ser totalmente desenvolvido voltado com base no usuário final que vai utilizar o sistema.

**2.3 – VIABILIDADE**

A tecnologia para que o projeto da calculadora e totalmente viável será feito com a linguagem de programação java orientado a objetos.

**2.4 – PRAZO**

O prazo foi analisado pelo nossa equipe e está dentro do possível para ser feito.

**2.5 - REGRAS DE NEGOCIO**

Atualmente as calculadoras são feitas com interfaces pouco polidas e às vezes até um pouco difíceis de entender.

Nesse Projeto da calculadora ela será feita de forma mais simplificada com um design mais clean sem muitas distrações para o usuário final pode ter agilidade no processo de calcular operações matemáticas.

**3 – PROBLEMÁTICA**

O uso da calculadora poderá provoca uma redução no cálculo escrito e mecanizado. Mas será socialmente preocupante um aluno não encontrar, com a mesma rapidez que um estudante há 20 anos, o quociente de um número de sete algarismos por quatro algarismos, utilizando unicamente lápis e papel? A destreza de cálculo numa situação como esta, contribuirá para reforçar a compreensão da operação? Acreditamos que não. No entanto, poderá ser preocupante se o aluno, observando o dividendo e o divisor, não conseguir ter mentalmente uma ordem de grandeza do quociente. Aliás, temos visto que há muitos alunos que nunca usaram a calculadora, para os quais 3 - 1 = 2!

Não é preocupante a diminuição do cálculo escrito e das técnicas tradicionais, pois o uso da calculadora de forma consciente traz implícito o desenvolvimento do cálculo mental e da estimativa.

Perde-se em habilidades mecânicas, mas é possível ampliar a compreensão da realidade dos números, do seu sentido na vida e nos problemas e da sua ordem de grandeza.

Desenvolver o sentido de número e capacidades como o cálculo mental e a estimativa são objetivos da aritmética básica que ficam extremamente valorizados com a introdução da calculadora. Isso porque consideramos que desenvolver um sentido sobre números é muito mais que fazer contas, é construir uma rede de ideias, esquemas e operações conceituais que levem o aluno a utilizar os conceitos em uma ampla variedade de situações.

**4 – ESCOPO**

Sua interface é muito simples o que facilita a sua compreensão e utilização. O aplicativo lembra-se de sua última operação, desligando ou não a calculadora. Você pode, em seguida, visualizá-los ao fazer suas contas, por exemplo. Também é possível voltar atrás, se você quiser fazer uma correção no seu cálculo, ou para armazenar seus cálculos, e até mesmo as funções matemáticas.

**5 – CICLOS DE VIDA DO SOFTWARE**

O modelo de desenvolvimento é essencial para o projeto de um software, pois com ele iremos definir as melhores ações no decorrer do seu ciclo de vida.

De acordo com a (NBR ISO/IEC 12207:1998), o ciclo de vida é a estrutura contendo processos, atividades e tarefas envolvidas no desenvolvimento, operação e manutenção de um produto de software, abrangendo a vida do sistema, desde a definição de seus requisitos até o término de seu uso.

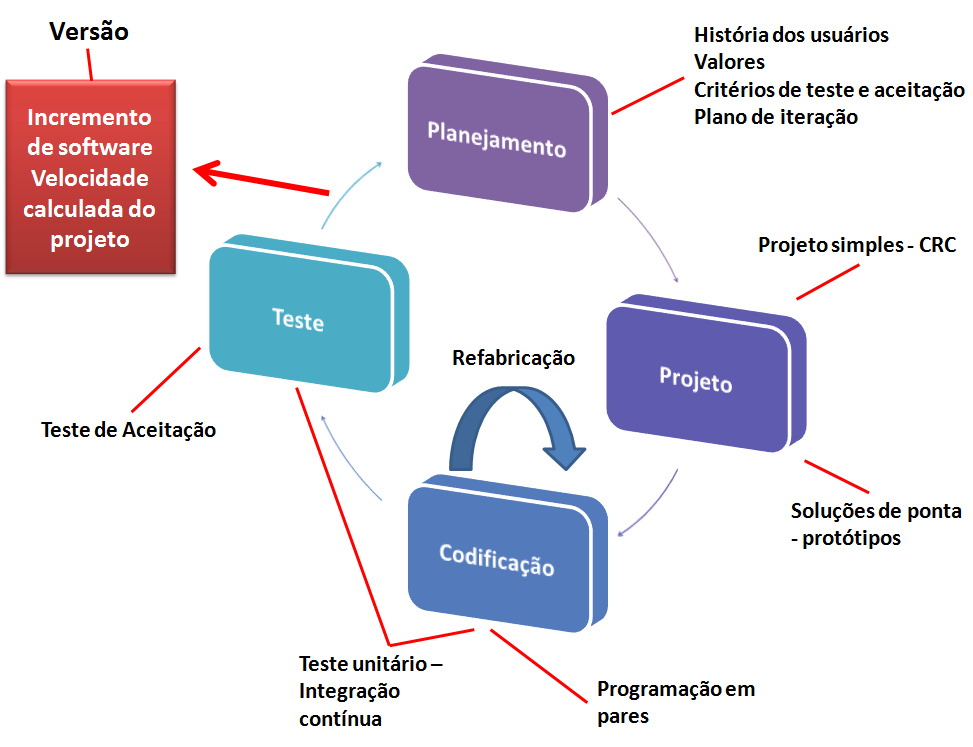


Figura 1: Processo XP.Fonte: PRESSMAN (2010).

O modelo de desenvolvimento escolhido foi o XP (Extreme Programming), por ser um processo robusto e bem estruturado em suas fases. A interação entre usuário e desenvolvedor é essencial para o sucesso e grande ciclo vida do projeto.

Com o objetivo de fazer uma aplicação simples e pratica para o uso no cotidiano, temos que contar com o feedback constante do nosso usuário. Assim podemos fazer constantes melhorias tanto visualmente como no código em si.

**6 – DIAGRAMAS UML**

**6.1 – DIAGRAMA DE CASO DE USO**

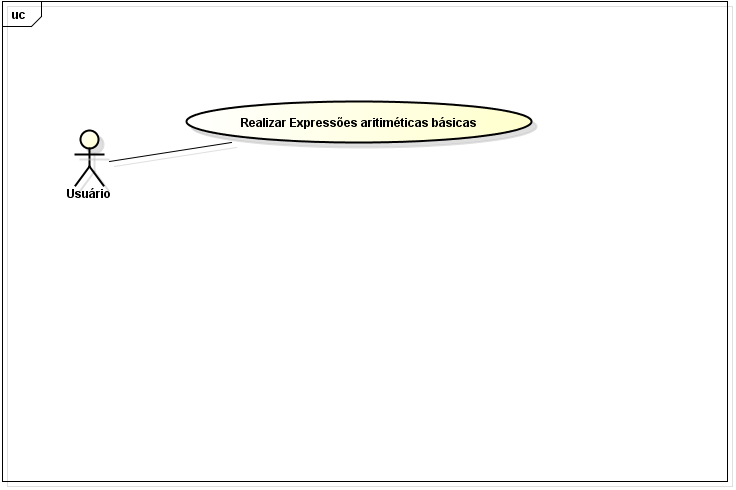


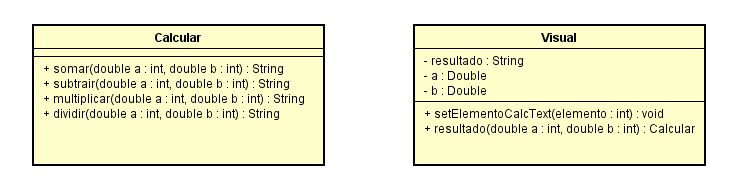
Figura 2 – Diagrama de Caso de Uso

**6.2 – LEVANTAMENTO DE CLASSES**

A partir das informações do diagrama de caso de uso, é possível: Desenvolver o diagrama de classes. Os componentes estruturais para um projeto de sistema orientado a objetos são as classes, assim fazendo de início um levantamento das classes do projeto separadamente.

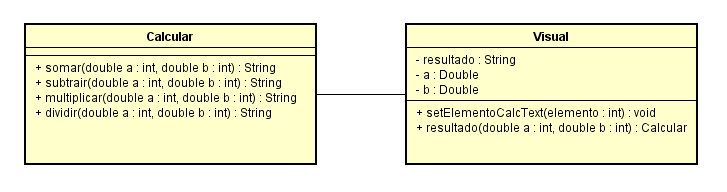
Onde temos o levantamento das classes, sendo elas a classe Calcular e a classe Visual, onde podemos observar todos os seus métodos e atributos. Sendo a visual responsável pela estrutura visual passada ao usuário e tendo como atributos os valores definidos por ele e os métodos capazes de passar esses mesmos valores a classe Calcular onde é feito todo o cálculo e retornado ao visual.

Figura 3 – Levantamento de Classes



**6.3 – DIAGRAMA DE CLASSES**

O Diagrama de Classe apresenta como as classes interagem entre si e qual a responsabilidade de cada classe na realização das operações solicitadas pelos atores, e uma representação da estrutura e relações das classes que servem de modelo para os objetos.

****Figura 4 – Diagrama de Classes

**6.4 – DIAGRAMA DE SEQUÊNCIA**

Consiste em um diagrama que tem o objetivo de mostrar como as mensagens entre os objetos são trocadas no decorrer do tempo para a realização de uma operação.

Como mostrado no diagrama, a troca de mensagens feitas pelo usuário e a aplicação contendo o trajeto de informações passadas e retornadas em suas determinadas etapas.

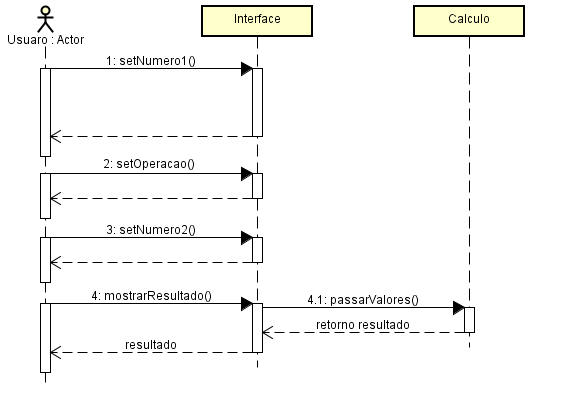
****

Figura 5 – Diagrama de Sequência

**6.5 – DIAGRAMA DE COMPONENTE**

O diagrama de componentes mostra a estrutura do sistema de software, que descreve os componentes do software, suas interfaces e suas dependências. É possível utilizar diagramas de componentes para modelar sistemas de software em um alto nível ou para mostrar componentes em um nível de pacote mais baixo.

No diagrama a seguir é mostrado a única tela que é composta a calculadora, onde podemos nela fazer todas as funcionalidades do sistema.

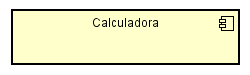
****

Figura 6 – Diagrama de Componente

**6.6 – DIAGRAMA DE ESTADO**

Em um diagrama de estado possui um comportamento e um estado. O estado de um objeto depende da atividade na qual ele está processando. Assim o diagrama mostra os possíveis estados de um objeto e as transações responsáveis pelas suas mudanças de estado.

Assim podendo ver neste diagrama os possíveis estados da calculadora, contendo três comportamentos e seus estados.

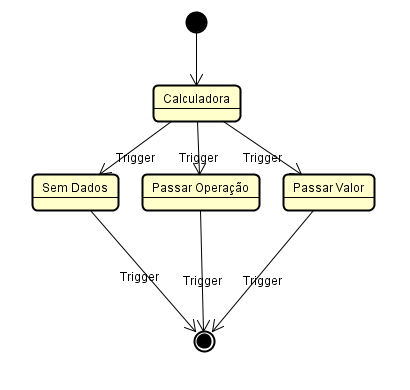
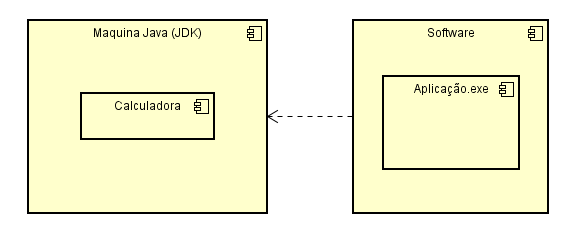
****

Figura 7 – Diagrama de Estado

**6.7 – DIAGRAMA DE IMPLANTAÇÃO**

O Diagrama de Implantação é o diagrama com a visão mais física da UML (GUEDES, 2007). Este diagrama foca a questão da organização da arquitetura física sobe a qual o software irá ser implantado e executado em termos de hardware, ou seja, as máquinas (computadores pessoais, servidores etc.) que suportam o sistema, além de definir como estas máquinas serão conectadas e por meio de quais protocolos se comunicarão e transmitirão as informações.

Figura 8 – Diagrama de Implantação

**6.8 – DIAGRAMAS DE ATIVIDADE**

O objetivo do diagrama de atividades é mostrar o fluxo de atividades em um único processo. O diagrama mostra como um atividade depende uma da outra. Tendo as atividades conectadas através de arcos (transições), que mostram as dependências entre elas.

Com o diagrama de atividade aplicado no desenvolvimento da calculadora, vemos as transições das atividades ao decorrer do sistema. Podendo visualizar as possíveis atividades da aplicação e suas ordens de processo.

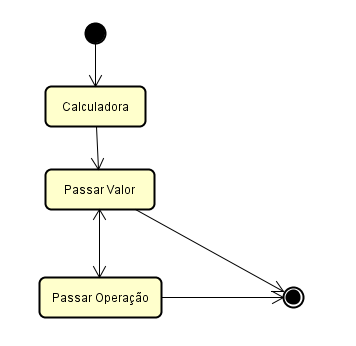
****

Figura 9 – Diagrama de Atividade

**7 – CONCLUSÃO**

Nos tempos de hoje a calculadora é uma ferramenta indispensável, ajudando o usuário a resolver cálculos com a maior agilidade e precisão. Concluindo que a aplicação deste software é gigantesca e indispensável em certos casos, sendo uma aplicação altamente implementável para maiores funcionalidades.

**REFERÊNCIAS**

BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. **UML: Guia do Usuário.** 12ª Ed., Rio de Janeiro: Campus, 2000.

COCKBURN, Alistair; **Escrevendo Casos de Uso Eficazes**; Bookman; 2007.

Criando Modelos e Diagramas, IBM. Disponível em: <<http://www-01.ibm.com/support/knowledgecenter/SS5JSH_9.1.1/com.ibm.xtools.modeler.doc/topics/cdepd.html?lang=pt-br>>. Acesso em 2 de novembro de 2015.

JONES, M.P; **Fundamentos do Desenho Orientado a Objetos com UML**; Makron Books; 2001.

KOBRYNM, Booch, Jacobson, Rumbaugh. UML Essencial: **Um breve guia para linguagem-padrão de modelagem de objetos**; Bookman, 2005.

SOMMERVILLE, Ian. **Engenharia de Software**. 8ª Ed., São Paulo: Addison Wesley Brasil, 2007.

LARMAN, G. **Utilizando** **UML e padrões: Uma introdução à análise e ao projeto orientados a objetos**; Tradução Luiz A Meirelles Salgado. Bookman Porto Alegre, 2000.

Portal Educação, Universidade Federal de Campina Grande. Disponível em: <<http://www.dsc.ufcg.edu.br/~sampaio/cursos/2007.1/Graduacao/SI-II/Uml/diagramas/atividades/diag_atividades.htm>>. Acesso em 1 de novembro de 2015.

Portal Educação, Universidade Federal de Campina Grande. Disponível em: <<http://www.dsc.ufcg.edu.br/~sampaio/cursos/2007.1/Graduacao/SI-II/Uml/diagramas/estado/diag_estados.htm>>. Acesso em 1 de novembro de 2015.

PRESSMAN, Roger S. **Engenharia de software.** 6ª ed. Porto Alegre: Bookman, 2006.