SACE

Sistema de Apoio ao Controle de Estoque

Documento de Arquitetura

Jairo Charnoski

Janisson Gois

Sumário

[1- Introdução 3](#_Toc260344803)

[1.1- Finalidade 3](#_Toc260344804)

[1.2- Escopo 3](#_Toc260344805)

[1.3- Referencias 3](#_Toc260344806)

[2- Representação da arquitetura 3](#_Toc260344807)

[3- Metas e restrições de arquitetura 3](#_Toc260344808)

[4- Visão de caso de uso 4](#_Toc260344809)

[4.1- realizações de casos de uso 4](#_Toc260344810)

[5- Visão lógica 5](#_Toc260344811)

[5.1- Visão geral 5](#_Toc260344812)

[5.2- Pacotes de designs significativos do ponto de vista da arquitetura 5](#_Toc260344813)

[6- Visão de processos 6](#_Toc260344814)

[7- Visão de implantação 6](#_Toc260344815)

[8- Visão de implementação 6](#_Toc260344816)

[8.1 – visão geral 6](#_Toc260344817)

[8.2- camadas 8](#_Toc260344818)

[9- Visão de dados 9](#_Toc260344819)

[10- Tamanho e desempenho 10](#_Toc260344820)

[11- Qualidade 10](#_Toc260344821)

1. Introdução
   1. Finalidade

O documento de arquitetura tem a finalidade de fornecer uma visão arquitetural do sistema (SACE), usando as visões de arquitetura mencionadas nesse documento para representação dos aspectos do sistema, a fim de ajudar a capturar e comunicar a definição das decisões arquiteturais.

* 1. Escopo

Este documento de Arquitetura de Software se aplica ao SACE, que será desenvolvido pelos alunos Jairo e Janisson do curso de *Sistemas de Informação da Universidade Federal de Sergipe*, o qual é projeto da disciplina de Engenharia de Software II.

* 1. Referencias

Casos de uso do sistema SACE:

* CSU01- Manter pessoas
* CSU02 – Vendas ECF
* CSU03 – Relatório ECF
* CSU04 – Controle de acessos
* CSU05 – Converter receitas
* CSU06 – Manter Usuários

Outros documentos:

* Documento de arquitetura SIGA
* Projeto.uml SACE

1. Representação da arquitetura

Este documento apresenta a arquitetura a partir da visão de caso de uso. Essa visão é apresentada como Modelos do *StarUML* e utiliza a Linguagem Unificada de Modelagem (*UML – Unified Modeling Language*).

1. Metas e restrições de arquitetura

Os requisitos de sistema de maior prioridade estando diretamente relacionados com a arquitetura são:

* Será uma aplicação Desktop.
* Desenvolvida em .Net C#.
* Banco de dados MySQL.

1. Visão de caso de uso

- Manter pessoas

-Manter usuários

-Controle de acesso

-Vendas ECF

-Relatório ECF

-Converter Receitas

## 4.1- realizações de casos de uso

-Diagrama de caso de uso:



-Descrição dos casos de uso:

\* Manter pessoas: Controle de pessoas cadastradas no sistema ou que serão cadastradas.

\*Manter usuários: Cadastra e mantém usuários do sistema com seus perfis.

\*Controle de acesso: Permite ou restringe determinados perfis de usuários de acessarem determinadas telas.

\*Vendas ECF: Controle sobre a impressão do cupom fiscal da venda e da baixa da venda no estoque.

\*Relatório ECF: Relatório controle da venda mensal.

\*Converter Receitas: Cadastra receitas, e converte produtos de acordo com as receitas

1. Visão lógica

## 5.1- Visão geral

A visão lógica do sistema SACE é composta por dois pacotes principais e um de apoio, são eles:

Interface: é a camada de apresentação, ou seja, de interação com o usuário.

Apoio: é a camada onde ficam os componentes uteis para o sistema, assim como a parte de negocio.

Dados: é a camada de acesso aos dados, na qual está diretamente responsável pela interação com a persistência dos dados.

## 5.2- Pacotes de designs significativos do ponto de vista da arquitetura



1. Visão de processos

<Ainda não definida>

1. Visão de implantação



1. Visão de implementação

8.1 – visão geral

O refinamento das classes de análise para os componentes de projeto será descrito passo a passo a seguir:

As classes de fronteira caso sejam telas que interaja com os usuários, será renomeada iniciando pela sigla “Frm” seguida pelo nome da classe de análise sem a palavra tela e sem o estereótipo, como por exemplo:

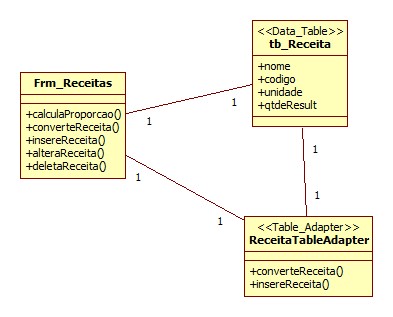
 

Por ter apenas 2 camadas os métodos das classes de controle passarão para a camada de dados caso esses métodos manipulem dados, caso contrario passarão para a classe correspondente na camada de fronteira. Como no exemplo abaixo:

As classes de entidade e entity collection serão convertidas em um único TableAdapter com o mesmo nome da entidade correspondente com o prefixo “tb\_”, com os métodos da entity collection e os atributos da entidade, como mostrado no exemplo abaixo:





8.2- camadas



1. Visão de dados

<Ainda não definida>

1. Arquitetura de exceções

O sistema terá classes de tratamento de exceção especificas para os prováveis tipos de erros que ocorrem numa determinada camada, o diagrama abaixo mostras as classes a serem implementadas em vermelho:

1. Tamanho e desempenho

<Ainda não definida>

1. Qualidade

<Ainda não definida>