SCC

Sistema de Controle de Contas

Documento de Arquitetura

Jairo Charnoski

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

Este documento tem a finalidade de fornecer uma visão arquitetural do sistema, usando as visões de arquitetura mencionadas nesse documento para representação dos aspectos do sistema, com a intenção de ajudar capturar e transmitir as decisões significativas da arquitetura tomadas em relação ao sistema.

* 1. Escopo

Este Documento de Arquitetura de Software se aplica ao Sistema de Controle de Contas, que será desenvolvido pelo aluno Jairo Charnoski do Nascimento, do curso de Sistemas de Informação da Universidade Federal de Sergipe, sendo este um projeto da disciplina de Engenharia de Software II.

* 1. Definições, Acrônimos e Abreviações

Todas as definições juntamente com os termos, acrônimos e abreviações necessárias à adequada interpretação e entendimento deste documento podem ser encontradas no documento de Glossário do Projeto, pode ser acessado através do endereço: http://code.google.com/p/controle-de-contas/source/browse/trunk/Modelagem\_Negocio/Modelo%20de%20Casos%20de%20Uso/ Glosário.doc.

* 1. Referências

Os seguintes documentos foram utilizados como referência para a elaboração do documento arquitetura:

* Casos de uso http://code.google.com/p/controle-de-contas/source/browse/trunk/Modelagem\_Negocio/Modelo%20de%20Casos%20de%20Uso/
* <http://www.wthreex.com/rup/portugues/webtmpl/templates/a_and_d/rup_sad.htm#1.3>

## Visão Geral

A seção 2 descreve o que a arquitetura do software representa para o sistema. A seção 3 descreve os requisitos de software, na seção 4 são listados os casos de uso que representam alguma funcionalidade significante para o sistema. As seções 5,6,7,8,9 apresentam respectivamente a visão lógica, de processo, de implantação, de implementação e de dados do sistema.

A seção 10 apresenta uma descrição das principais características de tamanho e performance que impactam a arquitetura, bem como as metas de performance desejadas. Finalmente, a seção 11 apresenta uma descrição de como a arquitetura de software contribui para a qualidade geral do sistema

1. Representação da arquitetura

Este documento apresenta a arquitetura como uma série de visualizações; visualização caso de uso, visualização lógica, visualização do processo e visualização da implementação. Essas visões são apresentadas como Modelos do Rational Rose e utilizam a Linguagem Unificada de Modelagem (UML).

O estilo arquitetural a ser utilizado no desenvolvimento do projeto será o de camadas, onde cada camada pode ser composta por vários componentes que interagem entre si ou entre os componentes de camadas adjacentes. Serão utilizados também outros estilos arquiteturais, tais como: Arquitetura baseada em Componentes e Client-Server.

1. Metas e restrições de arquitetura

Existem algumas restrições de requisito não-funcionais e de sistema principais que têm uma relação significativa com a arquitetura.

São elas:

• Será uma aplicação Web;

• A plataforma de desenvolvimento será a .NET utilizando C# como linguagem de desenvolvimento;

•O Sistema Gerenciador de Banco de Dados escolhido para suportar a aplicação será o SQLServer 2008.

1. Visão de caso de uso

CSU01- Contas a pagar

CSU02 – Contas a receber

CSU03 – Cadastro de Fornecedores

CSU04 – Movimentação de Contas

CSU05 – Usuários

4.1- realizações de casos de uso

-Diagrama de caso de uso:



-Descrição dos casos de uso:

\* Contas a pagar: Controle de contas a pagar cadastradas no sistema ou que serão cadastradas.

\*Contas a receber: Controle de contas a receber cadastradas no sistema ou que serão cadastradas.

\*Cadastro de fornecedores: Controle de fornecedores cadastrados ou que serão cadastrados.

\*Movimentação de contas: Controle sobreas contas bancárias do usuário.

\*Usuários: Controle sobre os usuários do sistema.

1. Visão lógica

## 5.1- Visão geral

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

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

Negócio: é a camada onde ficam os componentes úteis para o sistema, assim como a parte de negócio.

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 dados
2. Tamanho e desempenho

Estruturando o sistema em camadas e a programação através de classes, objetivamos obter uma boa performance do sistema e trataremos a programação para que não haja processamentos desnecessários. Segue a descrição das principais características de tamanho e desempenho do sistema:

O sistema deve permitir o acesso simultâneo aos usuários restritos ao sistema e a operação deverá ser continuada e ininterrupta.

O tempo de resposta para carregar uma página não deve exceder um minuto sob condições normais.

1. Qualidade

Segue a descrição de como a arquitetura de software contribui para a qualidade geral do sistema:

Portabilidade: a fim de garantir portabilidade e independência de plataforma, o sistema deve ser desenvolvido na plataforma Java.

Modularidade: o sistema deve ser desenvolvido em camadas, havendo uma interface de comunicação bem definida entre as mesmas.

Manutenibilidade: O sistema deve adotar padrões de documentação e codificação bem definidos. O código fonte do sistema deve estar bem documentado, ser robusto e fácil de debugar e manter.

Reusabilidade: a arquitetura do sistema deve ser tal que permita a utilização de classes e componentes de outros projetos, favorecendo o tempo de produção e a qualidade do produto gerado.

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:

