Concepta

Documento de Arquitetura de Software

Versão 1.0

Histórico de Revisões

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versão** | **Descrição** | **Autor** |
| 18/04/09 | 1.0 | Elaboração Inicial do Documento | lfol e prga2 |
| 28/04/09 | 1.1 | Finalização do Documento | prga2 |
|  |  |  |  |
|  |  |  |  |

Índice

1. Introdução 4

1.1 Finalidade 4

1.2 Escopo 4

1.3 Referências 4

1.4 Visão Geral 4

2. Representação da Arquitetura 4

3. Objetivos e Restrições da Arquitetura 5

4. Visão de Casos de Uso 5

4.1 Execução de Casos de Uso 5

5. Visão Lógica 5

5.1 Visão Geral 5

5.2 Pacotes significantes da arquitetura 6

5.2.1 CORE: 6

5.2.2 Teclado: 6

5.2.3 Mouse 7

5.2.4 Dicionário 7

5.2.5 Perfil 7

6. Visão da Implementação 7

6.1 Visão Geral 7

6.1.1 Visão de Subsistemas 7

6.1.2 Visão de Camadas 8

6.2 Camadas 8

6.2.1 Interface 9

6.2.2 Controle 9

6.2.3 Básica 9

6.2.4 Comunicação 9

7. Tamanho e Performance 10

8. Qualidade 10

Documento de Arquitetura de Software

# Introdução

## Finalidade

Este documento tem como finalidade prover uma visão geral da arquitetura do sistema, usando várias formas de abordagem. Tem como objetivo também mostrar como se organiza a implementação do sistema em vários níveis e traduz os requisitos do sistema para uma linguagem mais familiar ao desenvolvedor.

## Escopo

Este documento tem como função auxiliar na implementação do sistema como um todo, pois fornece uma visão de baixo nível do problema a que o sistema a ser desenvolvido visa resolver.

## Referências

Modelo de Design;

Modelo de Casos de Uso;

Guia de Programação;

Plano de Desenvolvimento de Software;

Especificação de Requisitos de Software.

## Visão Geral

O restante do documento está organizado da seguinte maneira:

2. Representação da Arquitetura: Esta seção descreve qual é a arquitetura de software do sistema atual e como ela é representada.

3. Objetivo e Restrições da Arquitetura: Esta seção descreve os requisitos de software e os objetivos que têm um impacto significativo na arquitetura, como proteção, segurança, privacidade, uso de um produto desenvolvido internamente e adquirido pronto para ser usado, portabilidade, distribuição e reutilização. Ela também captura as restrições especiais que podem ser aplicáveis: estratégia de design e implementação, ferramentas de desenvolvimento, estrutura das equipes, cronograma, código-fonte legado e assim por diante.

4. Visão da Implementação: Esta seção descreve a estrutura geral do modelo de implementação, a divisão do software em camadas e subsistemas no modelo de implementação e todos os componentes significativos do ponto de vista da arquitetura.

# Representação da Arquitetura

Teremos a representação da arquitetura em 3 visões diferentes:

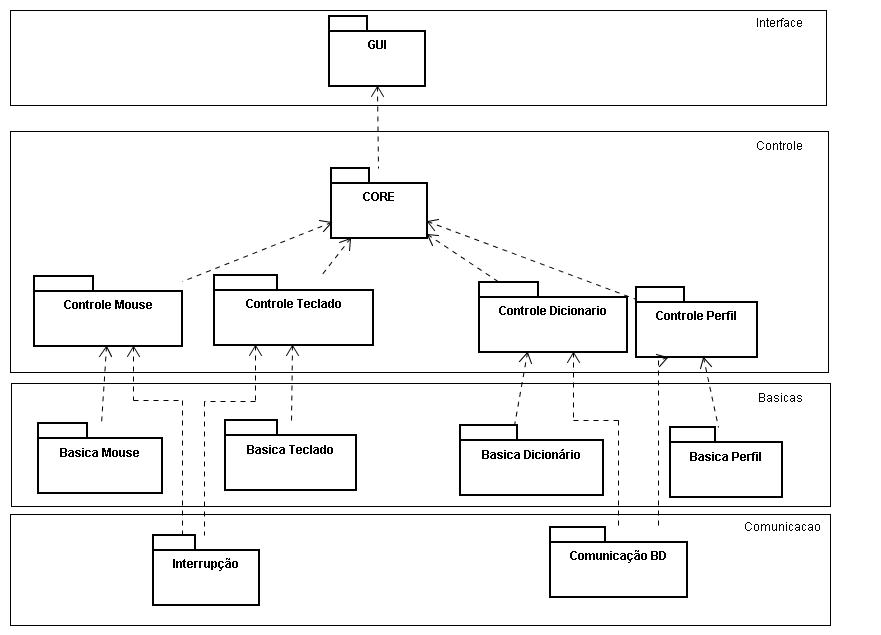
* Visão de Casos de Uso: Na visão de casos de uso nós temos os principais casos de usos do sistema mostrados. Os principais casos de uso são aquele que capturam as principais funcionalidades do sistema ou aqueles que capturam os requisitos mais críticos do sistema.
* Visão Lógica: Na visão lógica nós temos a apresentação da arquitetura do sistema em pacotes e camadas e a descrição dos pacotes mais significativos e suas classes.
* Visão de Implementação: Na visão de implementação nós temos a descrição mais aprofundada das camadas da arquitetura e que pacotes compõem cada camada explicando o funcionamento de cada um.

# Objetivos e Restrições da Arquitetura

O objetivo da construção da arquitetura do sistema é ganhar confiabilidade para o produto e reusabilidade. Ganhamos confiabilidade, pois com arquitetura definida a implementação se torna mais concisa e intuitiva para todos os desenvolvedores. A Reusabilidade pode ser facilmente empregada com uma arquitetura bem definida e fiel ao desenvolvimento do projeto.

# Visão Lógica

## Visão Geral



Na arquitetura do Concepta teremos, 4 camadas na arquitetura:

* A camada de GUI: onde teremos o pacote ligado à interface gráfica do sistema.
* A camada de Controle: onde teremos o CORE que é o controle geral da nossa aplicação e os controles menores que vão executar funcionalidades especificas como o controle do teclado e dicionário.
* A camada básica: onde teremos as classes básicas que definem os elementos básicos do nosso sistema como o elemento teclado, o elemento dicionário.
* A camada de comunicação: onde teremos as classes que promoveram a comunicação entre o sistema e os componentes externos ao sistema tais como: Sistema Operacional e Banco de Dados.

## Pacotes significantes da arquitetura

### CORE:

O CORE é responsável por controlar todos os controladores menores. Ele vai ordenar e fazer requisições aos controladores módulos menores, organizando o sistema.

### Teclado:

O pacote teclado tem todas as classes responsáveis pelo controle da emulação de teclado. O pacote teclado é chamado pelo pacote CORE que o controla. O teclado também faz as requisições as pacote interrupção para requisitar as interrupções desejadas ou para recebê-las.

### Mouse

O pacote mouse é responsável por controlar toda a emulação de mouse, contendo as classes que terão essa responsabilidade. Esse pacote vai ser controlador pelo CORE que fará requisições a este. O pacote mouse também fará requisições, assim como o teclado, ao pacote de interrupções que repassa essas ou devolve alguma interrupção do S.O.

### Dicionário

O pacote dicionário contem todas as classes necessárias para efetuar a funcionalidade de dicionário inteligente proposta pelo concepta. Esse pacote recebe requisições do CORE que o controla com estas. O pacote dicionário faz requisições ao pacote de Comunicação com B.D. para fazer a gestão de palavras do dicionário.

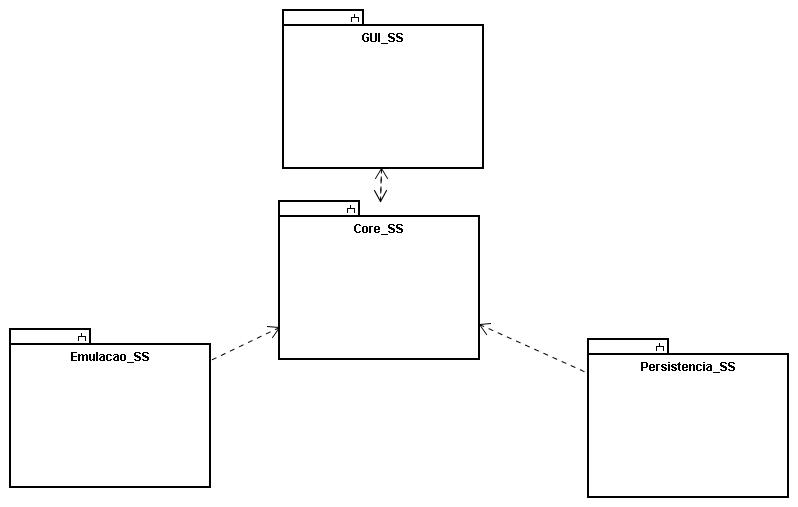
### Perfil

O pacote perfil tem as classes responsáveis pela gestão do perfil do usuário no sistema. Ele é requisitado pelo CORE para efetuar suas funções. O pacote perfil requisita o pacote comunicação B.D para gerir as informações sobro o usuário no Base de dados.

# Visão da Implementação

## Visão Geral

### Visão de Subsistemas

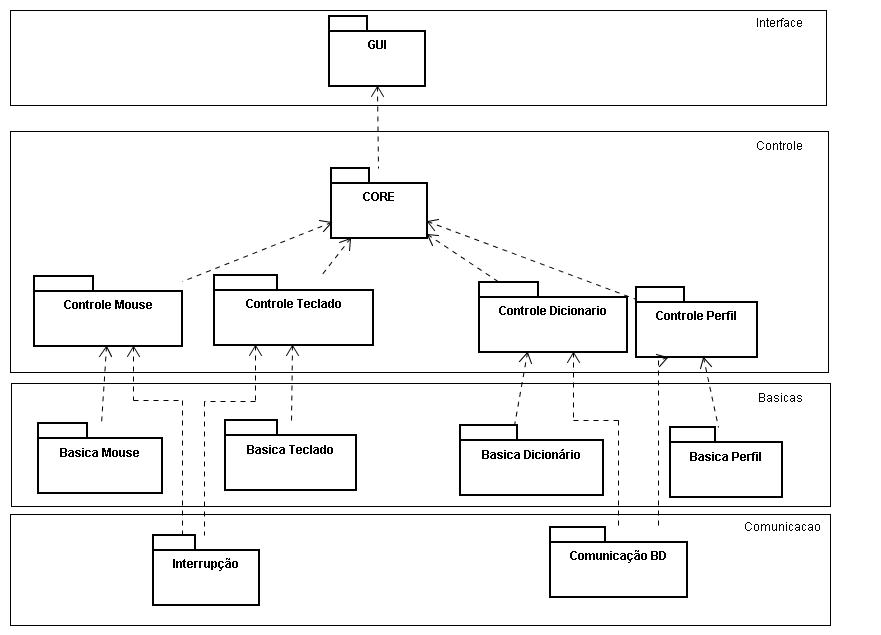


Temos a divisão em 4 subsistemas, divisão meramente funcional.

Onde temos os seguintes subsistemas:

* GUI: responsável pela interface gráfica do sistema
* CORE: responsável pelo controle geral do sistema
* Emulação: responsável pela parte de emulação de mouse e teclado.
* Persistência: responsável pela parte persistente do sistema que são dicionário e o perfil do usuário.

### Visão de Camadas



Nesta visão temos os pacotes organizados nas suas respectivas camadas que são:

* Interface: contem os pacotes que são responsáveis pela parte gráfica do sistema.
* Controle: contem os pacotes responsáveis pela parte de controle do sistema desde o controle geral do sistema, o CORE, até os controles dos módulos menores.
* Básica: contem os pacotes responsáveis por conter as classes básicas que são as classes que descrevem as entidades básicas do sistema e seu comportamento.
* Comunicação: contem os pacotes responsáveis pela comunicação entre sistema e o ambiente externo como: S.O. e o B.D.

O sistema terá o seguinte comportamento: a camada de controle controlará todo o sistema, utilizando as classes básicas para gerenciar o sistema. O camada de controle irá também se comunicar com o ambiente externo promovendo as funcionalidades do concepta que empregam um meio externo. Assim como o controle também passará informações para o engine gráfico que descrevera de acordo com as informações passadas a interface do sistema.

## Camadas

### Interface

Na camada de interface temos apenas um único pacote, o pacote de GUI que contém todas as classes responsáveis por apresentar o programa ao usuário. O pacote GUI recebe parâmetros do CORE controlador geral, para saber informações de estados do programa como, por exemplo, quando o usuário apertou uma tecla do teclado, temos que mudar um dado painel.

### Controle

Nessa camada temos o cérebro do programa, todos os controladores de cada funcionalidade do programa estão alocados nessa camada e nela temos 5 pacotes:

* CORE: é responsável pela organização e por controlar os módulos menores. Esse pacote vai criar objetos de todas as classes dos controladores menores para gerenciar todo o sistema com o auxilio das classes menores. Ele também vai receber atualizações da GUI para, por exemplo, saber onde o usuário está clicando.
* Controlador Teclado: é responsável por fazer a o controle da parte de emulação de teclado do sistema. Esse pacote é responsável por interpretar requisições do CORE e fazer requisições ao pacote de Interrupções para receber ou enviar as interrupções desejadas.
* Controlador Mouse: é responsável por fazer as todo controle de emulação de mouse do concepta. Esse pacote irá receber requisições do CORE para então requisitar ao pacote de Interrupção as interrupções recebidas pelo sistema ou as interrupções a serem enviadas para o sistema.
* Controlador Dicionário: Esse pacote é responsável pelo controle do teclado inteligente do sistema. Ele vai receber requisições do CORE, requisitando então o pacote de comunicação do Banco de Dados para armazenar ou requisitar dados da base do usuário.
* Controlador Perfil: Esse pacote fará o gerenciamento do perfil de usuário implementado pelo concepta. Esse classe receberá requisições do CORE pedindo ou enviando as configurações padrão do usuário e então o pacote requisitará ao pacote de Comunicação com o B.D. para armazenar ou recuperar as informações sobre a configuração do usuário para o sistema.

### Básica

Nessa camada nós temos os pacote que possuem as classes básicas que definem os elementos do nosso sistema. Definem os elementos básicos descrevendo, além do próprio elemento, o comportamento básico desse elemento. Esses pacotes são:

* Basica Teclado: Nesse pacote temos todas as classes básicas que definem os elementos básicos para emulação de teclado do sistema.
* Basica Mouse: Nesse pacote temos todas as classes básicas que definem os elementos básicos para emulação de mouse do sistema.
* Basica Dicionario: Nesse pacote temos todas as classes básicas que definem os elementos básicos para a implementação do Dicionário Inteligente do sistema.
* Basica Perfil: Nesse pacote temos todas as classes básicas que definem os elementos básicos para a implementação do Perfil de Usuário do sistema.

### Comunicação

Nessa camada nós temos os pacotes responsáveis por promover a comunicação entre o Concepta e o ambiente externo. Nesse caso temos duas possibilidades a comunicação com um banco de dados e uma comunicação com o sistema operacional e para isso temos 2 pacotes:

* Interrupção: o pacote de interrupção é responsável por promover a interação S.O. e Concepta. Esse pacote quando requisitado deve passar a interrupção do Concepta para o S.O. ou monitorar as interrupções do S.O. passando essa para o Concepta.
* Comunicação BD: esse pacote é responsável pela comunicação entre o Concepta e a base de dados do usuário. Esse sistema quando requisitado deve armazenar ou recuperar os dados da base de dados do usuário ativo.

# Tamanho e Performance

O tamanho do software concepta deve se manter praticamente constante sendo variável apenas a parte de banco de dados que deve ser variável porem dentro de uma margem pequena de espaço em disco.

A questão de performance do programa deve ser limitada pelo acesso do Concepta a base de dados. E esta por sua vez deve ser imperceptível para usuário tornando o concepta um programa agradável.

# Qualidade

Uma boa elaboração da arquitetura deve provocar uma integração bem sucedida do programa, o que trás como benefícios um padrão de qualidade maior tanto no quesito confiabilidade quanto no quesito manutenabilidade, característica fundamental no desenvolvimento de software.