DESENVOLVIMENTO DE UM SISTEMA DE GESTÃO DE ESTÁGIO

Gustavo Gomes Mendonça, Lucas Pereira Vasconcelos
Universidade Federal de Uberlândia, Faculdade de Engenharia Elétrica, Uberlândia – Minas Gerais,
{gustavo.gomes.mendonca, lucas.pereira.vasconcelos}@gmail.com

Resumo – Esse artigo busca aliar os conceitos acerca das linguagens de programação, frameworks, banco de dados e afins para descrever como foi implementado o sistema que contém as ferramentas necessárias para a gestão dos estágios obrigatórios dos alunos matriculados nos cursos de Graduação em Engenharia Elétrica e Biomédica oferecidos pela Faculdade de Engenharia Elétrica da Universidade Federal de Uberlândia.

***Palavras-Chave* – 3-tier, C#, NHibernate, Object-Relational Mapping, Programação Orientada a Objetos, software, Windows Presentation Fundation.**

DEVELOPMENT OF AN INTERNSHIP MANAGEMENT SYSTEM

Abstract – This article seeks to combine the concepts of programming languages, frameworks, databases and something like that to describe how the system which contains the tools necessary to the management of required internship of the students enrolled into the graduating courses in Electrical Engineer and Biomedical Engineer offered by the Faculty of Electrical Engineer of the Federal University of Uberlândia was implemented.

***Keywords* - 3-tier, C#, NHibernate, Object Oriented Programming, Object-Relational Mapping, software, Windows Presentation Fundation.**

I. INTRODUÇÃO

A necessidade de reimplementação do Sistema de Gestão de Estágios (SISE) utilizado pela secretaria da Faculdade de Engenharia Elétrica da Universidade Federal de Uberlândia, surgiu devido a dificuldade de manutenção do sistema usado até então. Foi proposta a criação de um novo software, que gerenciasse a situação do aluno quanto a seu estágio supervisionado, em uma arquitetura com bons níveis de manutenibilidade e usabilidade. O sistema foi feito com o intuito de facilitar tanto o cadastro quanto o acompanhamento do estágio do aluno, bem como a geração das fichas de matrícula e conclusão da disciplina de estágio supervisionado.

Este artigo discorre os métodos e técnicas adotados para o desenvolvimento, implementação e aplicação do sistema. Apresenta também os conceitos de desenvolvimento em camadas, banco de dados, frameworks para mapeamento objeto-relacional, padrões de projeto, arquitetura de software, além de vantagens e desvantagens da escolha de uma linguagem de programação, bem como dos elementos que a compõem.

X CEEL - ISSN 2178-8308
24 a 28 de setembro de 2012
Universidade Federal de Uberlândia - UFU
Uberlândia - Minas Gerais - Brasil

II. CORPO DO TRABALHO

A. Motivação do Desenvolvimento do Sistema

Com o desenvolvimento de tecnologias acerca de computação em geral, desenvolver um software orientado a objetos que já era difícil, se tornou ainda mais difícil já que a interação com o usuário aumentou e a complexidade das aplicações também. Um software eficiente deve ser estruturado de tal forma que classes sejam herdadas de maneira correta, observar corretamente a granularidade dos objetos dentro de cada classe, além de implementar interfaces de forma eficiente.

A aplicação desejada de certa maneira, além de ser específica para o problema explicitado, os problemas e soluções a cerca do desenvolvimento dessa devem ser de alguma forma ser aproveitados para futuros desenvolvimentos. A aplicação, ou seja, o software orientado a objetos ainda deve ser desenvolvido para que tenha o mínimo de falhas possíveis, para que evitem uma nova versão da aplicação, ou seja, um novo projeto para correção das falhas apresentadas.

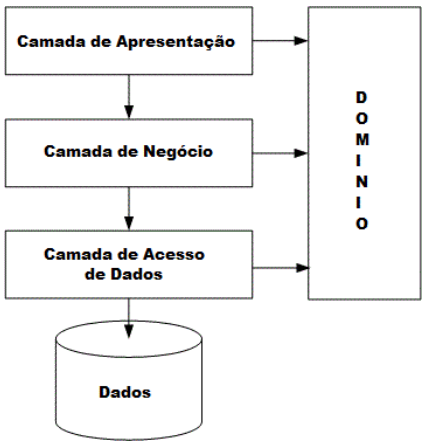

Fig. 1. Arquitetura em camadas.

Então, para que um projeto seja bem feito é preciso que seja bem modelado, o que não é fácil, além de poder surgir muitos erros. Pensando assim, um problema já resolvido uma vez pode ajudar no desenvolvimento de novos projetos e se as várias partes do projeto puderem ser implementadas de forma independente, a detecção de erros se tornaria mais fácil e localizada.

B. *Padrões de projeto*

Alguns problemas comummente encontrados durante o desenvolvimento de sistemas possuem soluções conhecidas e compartilhadas pela comunidade de desenvolvedores. Tais soluções, conhecidas como "Design Patterns" como ficaram populares após os autores Erich Gamma, Richard Helm, Ralph Johnson e John Vlissides conhecidos como "Gangue dos Quatro" (Gang of Four) ou simplesmente "GoF" publicarem em seu livro "Design Patterns: Elements of Reusable Object-Oriented Software" [1] um compilado de estratégias comuns na estruturação de sistemas de software.

Os padrões de projeto descritos em [1] são amplamente utilizados pela comunidade mundial de desenvolvedores. No software desenvolvido essa realidade não é diferente, ao analisar a estrutura do sistema, percebe-se claramente a utilização de vários desses padrões, como *proxies* ao instanciar objetos persistidos, *adaptors* na conversão de objetos em strings representativas e vice versa, *strategies* ao criar um método genérico para salvar ou atualizar registros no banco e *singletons* usados nos *Business Objects* e *Data Access Objects* do sistema.

C. *Desenvolvimento em Camadas*

Afim manter o código organizado e facilitar a reutilização e manutenção de código, o sistema foi dividido em três camadas principais: camada de acesso a dados, camada de negócio e camada de apresentação. Entretanto, os dados tratados em uma camada devem ser transferidos para as camadas camadas seguintes até alcançar seu destino final e, para que essa tarefa se realize com mais facilidade, foram criadas classes de domínio que representam as entidades do mundo real tratadas pelo sistema, como alunos, professores e estágios. Assim se for necessário, por exemplo, exibir uma lista de alunos persistida no sistema, basta carregar uma lista de objetos do tipo Aluno na camada de acesso a dados, valida-la segundo as regras de projeto na camada de negócio e repassar a lista para que a camada de apresentação possa arranjar e exibir os dados ao usuário. Tal hierarquia é exibida na Figura 1 [4].

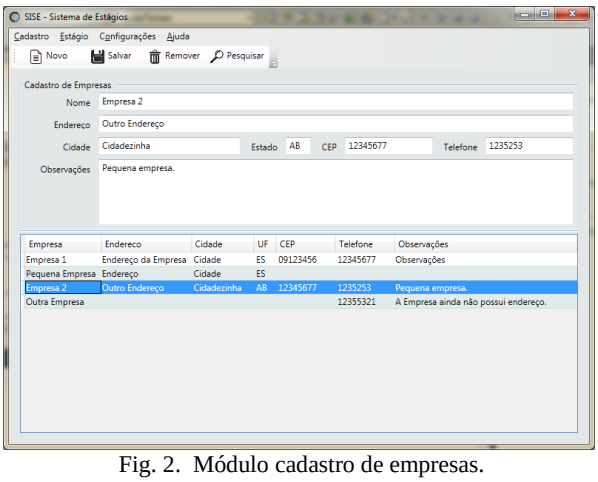

Fig. 2. Módulo cadastro de empresas.

A camada de acesso de dados é responsavel por acessar serviços de baixo nível do sistema como banco de dados, sistema de arquivos, etc. De forma mais específica, no sistema construído a camada de acesso é responsavel por manipular as conexões com o banco dados, converter os dados recuperados do banco em objetos do domínio do sistema e persistir em banco as informações relativas ao sistema.

A camada de negócio trata os dados coletados pela camada de acesso e provê interfaces de acesso à camada de apresentação. É na camada de negócio que as rotínas referentes às regras de negócio são definidas, ou seja, a camada de negócio contém toda a lógica funcional do software.

As informações presentes no sistema devem ser expostas ao usuário e a camada de apresentação é a responsável pela visualização desses dados. Nessa camada são definidos os elementos gráficos a serem apresentados e as vias de interação do usuário com o sistema.

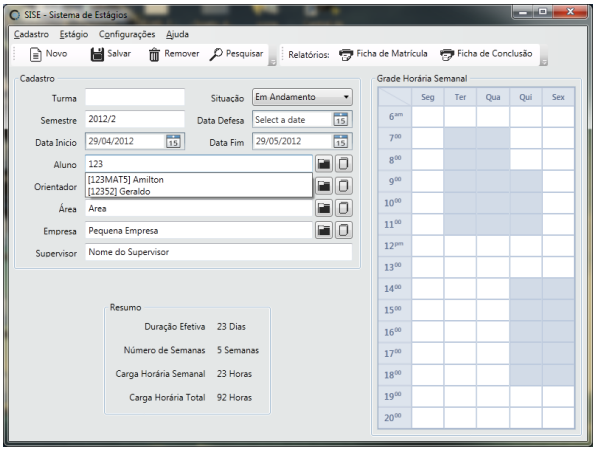

Fig. 3. Módulo responsável por cadastro de estágios.

D. *Modularização*

Pensando na alta manutenibilidade do sistema, o Sise foi separado em módulos onde cada módulo é responsável por um aspécto funcional do sistema. Esses módulos são separados em três categorias: cadastro estágio e configurações.

Como o próprio nome sugere, os módulos de cadastro são responsáveis pelo cadastro de um domínio específico. A Figura 2 exibe o módulo de cadastro de empresas.

O sistema contém dois módulos de estágio. O primeiro, ilustrado na Figura 3, é responsável por cadastrar novos estágios no sistema, retirar as fichas de matrícula e conclusão de estágios cadastrados e cálculo das grades horárias. Esse é o principal módulo do sistema e, devido a sua alta complexidade, a funcionalidade de pesquisa foi transferida para um segundo módulo.

A categoria de configurações contém três módulos: o módulo de usuário, responsável pelo cadastro de usuários e suas permissões no sistema, módulo de configurações gerais, que permite a alteração dos dados da instituição, e o módulo de configurações avançadas, que dá acesso direto a todos os parâmetros do sistema persistidos em banco, como pode ser Observado na Figura 4.

E. *Escolha da Linguagem C# e utilização do Windows Presentation Foudation*

Com o sistema já arquitetado, a linguagem orientada a objetos escolhida foi C# 4.0 para a implementação do

mesmo. À priori para criar aplicações em desktop para Windows o ".NET", que é o framework utilizado para rodar aplicações feitas em C#, oferece o Windows Forms Application ou WFA. O WFA foi criado juntamente com o framework .NET. Entretanto, apesar de sua popularidade, não foi utilizada nesse projeto.

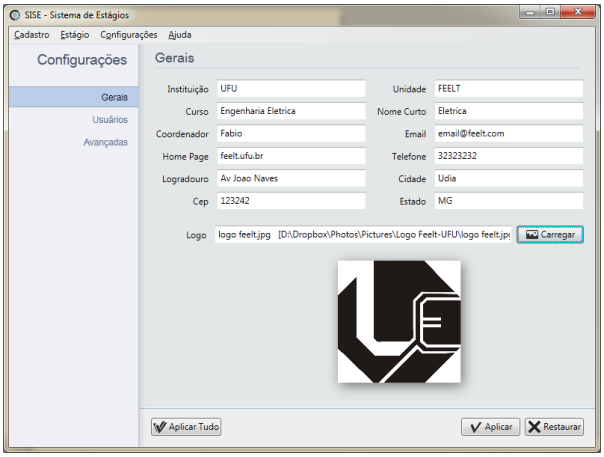

Fig. 4. Módulo responsável por cadastro de estágios.

A partir da necessidade de criar interfaces mais modernas, com mais funcionalidades e recursos, a miscrosoft decidiu criar o Windows Presentation Foudation ou WPF, ao invés de continuar com o Windows Forms. Logo, temos o primeiro motivo para o uso. A Microsoft, em conjunto com toda a comunidade .NET, está investindo no desenvolvimento do WPF e com o tempo teremos melhores recursos e mais inovação ao utilizá-lo. Conforme foi falado o Sise utiliza do paradigma arquitetura em três camadas, e o WFA não foi implementado para esse propósito, então precisamos de uma ferramenta que possua recursos para tal, e o WPF atende bem as especificações do projeto.

Além desses motivos que deixam clara a escolha para utilizar o WPF, ele ainda tem uma interface bonita, tem mecanismos para que o design da janela não se deforme ao redimensionar o tamanho da janela, já que com o advento das inovações tecnológicas surgem monitores com grandes resoluções de tela, e o sistema deve se ajustar a eles. Possui também suporte para mídias em geral, como vídeos, áudios, documentos, animações e até conteúdo 3D.

O WPF oferece criação de interfaces, assim é possível implementar seu projeto com diferentes tipos de interface. E para oferecer um melhor projeto de interfaces utiliza uma linguagem de marcação chamada XAML que permite uma separação entre o trabalho de design e o desenvolvimento do software em si, e atualmente é a maneira para se criar softwares recomendada pela Microsoft. Logo, foi adotado no projeto com a utilização de uma interface limpa, semelhante às interfaces web conforme mostrado na Figura 5 [3].

F. Escolha do Banco de Dados e aplicação do framework de persistência

Já que um sistema desses recebe muitos dados e precisa ser acessado simultaneamente por vários clientes, visto que o controle de estágios será feito por vários pessoas, é preciso criar um banco de dados que receba todos esses dados e os deixe disponíveis para consultas e correções. Ou seja, é necessário um sistema de banco de dados, nesse projeto foi adotado o MySQL para tal.

Para armazenar esses dados, várias tabelas são criadas para administrar esses dados no banco, para que essas tabelas sejam mapeadas e façam a comunicação com o sistema, se fez uso de uma ferramente chamada NHibernate. O NHibernate é um framework desenvolvido em .NET para persistir os objetos para bases de dados relacionais. Baseado em uma ferramenta de persistência de dados do Java, o Hibernate, tem a finalidade de persistir objetos .NET em banco de dados. O framework gera o código SQL necessário, certificando-se que os tipos e os valores são corretamente criados.

Dessa forma, o NHibernate faz o mapeamento entre a classe do projeto e a tabela do banco de dados, e como dito não há necessidade de codificar nenhum comando SQL, porém há necessidade de gerar um arquivo de mapeamento para cada tabela. Há ferramentas para auxiliar na criação desses arquivos, mas ainda sim, pode ser uma tarefa trabalhosa. Mas para um banco de dados pequenos, pode ser uma boa escolha e não somente para pequenos bancos visto que o Nhibernate já é um framework bastante utilizado e que de certa maneira facilita sim, a persistência do banco de dado [5].

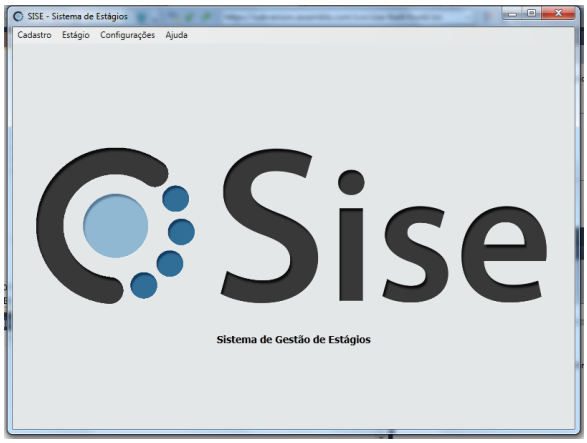

Fig. 5. Interface Sise.

Mas como dito, o NHibernate é um bom framework que possui uma boa funcionalidade, tanto é que é baseado no Hibernate do Java que tem uma boa popularidade, confiabilidade e eficiência. O porquê de não usar Java ao invés de C# pode surgir, mas sucintamente são linguagens bem parecidas baseadas em classes e orientação a objetos, mudam em questões de plataformas, modo de uso de bibliotecas e interfaces, mas de certa maneira que a escolha cabe muito à preferência do programador e da equipe. Dessa maneira o Sise foi implementado em C# com o mapeamento feito pelo NHibernate pelo banco de dados MySQL.

O esquema do projeto, como já comentado a maneira como foi arquitetado, pode ser observado na Figura 6. O "Sise.Wpf.DesktopApplication" como dito é a parte de interface do usuário, através dele são recebidos os dados e mostrados os dados requeridos pelo usuário. O "Sise.Wpf.Modules" possui os módulos que são carregados para o usuário, os controles que são implementados pelos módulos e são responsáveis pela comunicação da interface

do usuário com os pacotes que mapeiam, salvam e leem os dados do banco de dados, descrito por "Sise" na Figura 6.

G. Funcionamento do Software Sise

O software Sise tem por finalidade a Gerência de Estágios e como mostrado na Figura 5, na aba Cadastro o usuário poderá fazer cadastros dos Alunos, Áreas de atuação, Empresas Orientadores e dos certificados de estágio. Feito todo o cadastro acerca do aluno a aba Estágio faz a Gerência desses estágios, desde a carga horária do estágio, a situação do estágio, dias em que o aluno frequenta o estágio. Além de reunir todas as informações e gerencia-las, cria os relatórios de matrícula e conclusão do estágio para a impressão. A aba de Configurações traz três outras janelas, Configurações Avançadas, Gerais e de Usuário.

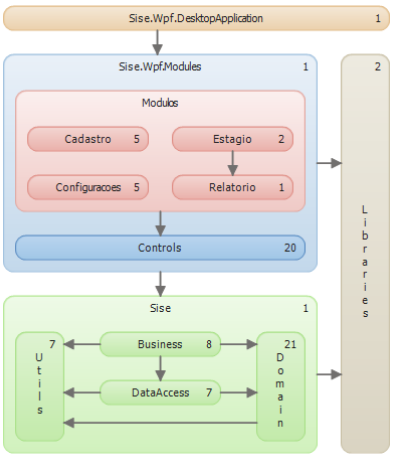

Fig. 6. Esquema do Projeto Sise.

As Configurações Avançadas trazem as informações que ficam diretamente no banco de dados, como nome do coordenador e da instituição, endereço entre outras informações e como pode ocorrer de serem mudadas foi feito esse tela. Como são informações muito importantes já que são impressas nos relatórios de estágio, tem acesso restrito. O acesso restrito é feito através da tela de Configuração de Usuário, onde o usuário com maior escalão pode limitar e liberar o acesso de cada tela. O usuário ao entrar no programa deve inserir um nome de usuário e uma senha para acessar o programa. A aba Configurações Gerais apresenta os parâmetros requeridos no relatório de gerência, para que sejam inseridos ou modificados. A aba Sobre contém informações acerca do projeto, para que se necessário o projeto tenha continuidade ou precise de melhorias.

Para a comunicação com o banco foi utilizado o NHibernate, para isso para cada tabela foi criado o arquivo de Mapeamento, o DAO (Data Access Object) de cada arquivo que seria para descoplar a camada de negócios da camada de persistência. Além disso foi criado o arquivo BO (Business Object) de cada tabela que são os objetos que estão relacionados à camada de uso da aplicação, ou seja, a camada de negócio.

Assim, o sistema foi arquitetado para que os relatórios de gerência de estágio sejam impressos com eficiência e com um bom controle. Além de ter um interface auto explicativa que auxilia o usuário, o sistema foi implementado para melhor atender a Coordenação dos cursos de Engenharia Elétrica e Biomédica da Faculdade de Engenharia Elétrica da Universidade Federal de Uberlândia.

III. CONCLUSÕES

O sistema de gerência de estágios ou Sise foi desenvolvido para organizar e deixar mais eficiente o cadastro e consulta dos estágios da Faculdade de Engenharia Elétrica da Universidade Federal de Uberlândia. O sistema atualmente já é utilizado na coordenação do curso a atende as exigências propostas para o desenvolvimento do mesmo.

O Sise ainda pode ser melhorado, e planeja-se que no primeiro ano de uso do software sejam corrigidos os erros e que sejam propostas melhorias e que essas mesmas sejam implementadas. O sistema possui as características de uma aplicação feita para fins profissionais e através dos estudos e do tempo dedicado para o desenvolvimento foi proporcionado ensinamentos sobre os conceitos explicitados nesse relatório, cumprimento de prazos, ou seja, o aprendizado de como criar um sistema profissional completo.

IV. REFERÊNCIAS BIBLIOGRÁFICAS

[1] E. Gamma, R. Helm, R. Johnson, J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, M. C. Eachers's 'Swans', Cerdon Art, Brown, Holland, 1998.

[2] H. Deitel, P. Deitel, K. Steinbuhler, *C# - Como Programar*, 1ª Edição, São Paulo, Pearson, 2007.

[3] J. Smith, *WPF vs. Windows Forms*. Acedido em 01 de Maio de 2012, em: http://joshsmithonwpf.wordpress.com/

[4] A. J. Baptistella, *Abordando a arquitetura MVC, e Design Patterns: Observer, Composite, Strategy*. Acedido em 01 de Maio de 2012, em: http://www.linhadecodigo.com.br

[5] Documentação do framework NHibernate. Acedido em 01 de Maio de 2012, em: http://nhforge.org/