



Allan Araujo Silva

C2D - Módulo de Credenciamento e Classificação de Docentes do Sistema Marvin

Vitória, ES

2017

Allan Araujo Silva

C2D - Módulo de Credenciamento e Classificação de Docentes do Sistema Marvin

Monografia apresentada ao Curso de Ciência da Computação do Departamento de Informática da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Bacharel em Ciência da Computação.

Universidade Federal do Espírito Santo – UFES

Centro Tecnológico

Departamento de Informática

Orientador: Prof. Dr. Vítor E. Silva Souza

Vitória, ES

2017

Allan Araujo Silva

C2D - Módulo de Credenciamento e Classificação de Docentes do Sistema Marvin

Monografia apresentada ao Curso de Ciência da Computação do Departamento de Informática da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Bacharel em Ciência da Computação.

Trabalho aprovado. Vitória, ES, 22 de dezembro de 2017:

Prof. Dr. Vítor E. Silva Souza
Orientador

Prof^a. Dr^a. Renata S. S. Guizzardi
Universidade Federal do Espírito Santo

Cássio Capucho Peçanha
Discente do PPGI/UFES

Vitória, ES
2017

Agradecimentos

Primeiramente gostaria de agradecer a Deus por ter me sustentado até aqui e por tudo o que Ele representa em minha vida. Sem Ele eu não teria chegado até aqui.

Sou imensamente grato pela família que tenho. Meus pais e minha irmã, Adalberto, Marta e Aléxia, que me acompanharam a todo momento. Agradeço a eles por todo o amor, dedicação, companheirismo, carinho e pelos ensinamentos que contribuíram para minha formação pessoal. Apesar das dificuldades, sempre me motivaram a buscar meus objetivos e correr atrás dos meus sonhos. Dedico minha graduação primeiramente a vocês.

Ao longo do curso tive ao meu lado muitos amigos e fiz novas amizades que desejo levar por toda a vida. Aos meus amigos, Vanderlei, Gabriel, Marcus, Daniel, Eduardo, Celso e Rafael, companheiros de cursos, agradeço por toda ajuda e troca de conhecimento, pelas tardes de estudos em plena cantina do CT e pelos bons momentos da faculdade. Agradeço a minha namorada, pelo seu zelo e amor, quem esteve ao meu lado me motivando a prosseguir e dando forças, por compreender os momentos de ausência devido aos estudos.

Não poderia deixar de agradecer aos professores e mestres que estiveram presentes em minha caminhada na graduação. Sem os ensinamentos deles e suas orientações, não teria aprendido conceitos técnicos e fortalecido meus pensamentos para minha formação acadêmica. Em especial, agradeço ao meu orientador, sempre me direcionando da melhor forma, tirando dúvidas, para que este trabalho fosse desenvolvido de forma satisfatória.

Foram 5 anos de muito aprendizado, esforço e dedicação, dificuldades e conquistas. A vida é assim. Não conseguimos avançar sem que tenhamos ajuda de alguém. E é por isso que sou grato. A todos que participaram de alguma forma desta etapa de minha vida, muito obrigado.

“Cada vez que você faz uma opção está transformando sua essência em alguma coisa um pouco diferente do que era antes.”

(C. S. Lewis)

Resumo

O Programa de Pós-Graduação em Informática da Universidade Federal do Espírito Santo (PPGI/UFES) necessita de um sistema/módulo para manipular dados de docentes, como produções bibliográficas — publicações de artigos científicos em diversos tipos de veículos (conferências e periódicos) —, pois atualmente a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) realiza avaliações dos PPGs de acordo com tais informações. Em consequência disso, o PPGI utiliza estes mesmos indicadores de produção como critério de credenciamento e credenciamento de docentes no programa. Assim, a proposta de um módulo vem para otimizar o processo de extração de dados dos docentes e automatizar o processo de credenciamento e classificação destes docentes no PPGI/UFES.

No desenvolvimento deste módulo, foram seguidos os processos de Engenharia de Software realizando levantamento de requisitos, especificação de requisitos, definição da arquitetura do sistema, implementação e testes. Foram colocados em prática os conhecimentos adquiridos em disciplinas realizadas no decorrer do curso, tais como Engenharia de Software, Engenharia de Requisitos, Projeto de Sistema de Software, Análise de Objetivos, Programação Orientada a Objetos e Desenvolvimento Web e Web Semântica. Também foram utilizadas técnicas de modelagem e desenvolvimento de sistemas orientado a objetos, destacando o método FrameWeb para projeto de aplicações Web centradas em *frameworks*.

Palavras-chaves: Sistemas Web, Engenharia de Software, Engenharia de Requisitos, Projeto de Sistemas, Java EE, Análise de Objetivos, FrameWeb

Lista de ilustrações

Figura 1 – Processo de Engenharia de Requisitos adaptado de (KOTONYA; SOM-MERVILLE, 1998).	14
Figura 2 – Construtos da linguagem iStar 2.0.	16
Figura 3 – Representação da Arquitetura de Rede <i>Cliente-Servidor</i>	20
Figura 4 – Representação da Arquitetura de Software <i>Model-View-Controller</i>	21
Figura 5 – Modelo de Objetivos do Módulo C2D.	27
Figura 6 – Diagrama de Casos de Uso do Módulo C2D.	28
Figura 7 – Diagrama de Classes do Módulo C2D.	30
Figura 8 – Modelo de pacotes para o sistema Marvin (LIMA, 2015).	31
Figura 9 – Pacotes na Implementação do C2D.	33
Figura 10 – Estrutura de páginas do sistema Marvin.	33
Figura 11 – Modelo de Entidades do módulo C2D.	34
Figura 12 – Classe <i>BaseDAO</i> e principais métodos.	35
Figura 13 – Modelo de Persistência do módulo C2D.	36
Figura 14 – Modelo de Navegação CRUD (SALVATORE, 2016).	37
Figura 15 – Modelo de Navegação para fluxo de cálculo de pontuação.	37
Figura 16 – Modelo de Navegação para fluxo de Extrair Informação de Qualificação.	38
Figura 17 – Classe <i>CrudServiceBean</i> e métodos CRUD.	39
Figura 18 – Modelo de Aplicação do sistema C2D, primeira parte.	40
Figura 19 – Modelo de Aplicação do sistema C2D, segunda parte.	40
Figura 20 – Tela inicial do sistema C2D.	41
Figura 21 – Tela de login do sistema C2D.	42
Figura 22 – Funcionalidades já existentes no Marvin.	42
Figura 23 – Funcionalidades implementadas para o C2D.	43
Figura 24 – Upload Lattes CV - C2D.	43
Figura 25 – Upload Lattes CV - C2D Aviso.	44
Figura 26 – Upload Qualis Data - C2D.	44
Figura 27 – Upload Qualis Data - C2D - erro de parsing.	45
Figura 28 – Upload Qualis Data - C2D - Journal.	45
Figura 29 – Upload Qualis Data - C2D - Confirmação.	46
Figura 30 – Create New Score System - C2D.	46
Figura 31 – Calculate Academics Score - C2D.	47
Figura 32 – Calculate Academics Score - C2D - Resultado.	50
Figura 33 – Calculate Academics Score - C2D - Detalhes.	50

Lista de abreviaturas e siglas

API	<i>Application Program Interface</i>
CDI	<i>Context and Dependency Injection</i>
CRUD	<i>Create, Read, Update and Retrieve</i>
DAO	<i>Data Access Object</i>
HTML	<i>HyperText Markup Language</i> - Linguagem de Marcação de Hipertexto
HTTP	<i>HyperText Transfer Protocol</i> - Protocolo de Transferência de Hipertexto
IDE	<i>Integrated Development Environment</i> - Ambiente de Desenvolvimento Integrado
JPA	<i>Java Persistence API</i>
JSF	<i>Java Server Faces</i>
JVM	<i>Java Virtual Machine</i>
MVC	<i>Model-View-Controller</i>
ORM	<i>Object-Relational Mapping</i> - Mapeamento Objeto-Relacional
PDF	<i>Portable Document Format</i>
SGBD	Sistema Gerenciador de Banco de Dados
SQL	<i>Structured Query Language</i>
URL	<i>Uniform Resource Locator</i> - Localizador Uniforme de Recursos
UML	<i>Unified Modeling Language</i> - Linguagem de Modelagem Unificada
XHTML	<i>EXtensible HyperText Markup</i>
XML	<i>RXtensible Markup Language</i>

Sumário

1	INTRODUÇÃO	10
1.1	Objetivos	10
1.2	Metodologia	11
1.3	Organização do Texto	12
2	REFERENCIAL TEÓRICO	13
2.1	Engenharia de Software	13
2.1.1	Engenharia de Requisitos	14
2.1.2	Análise de Objetivos	15
2.1.3	Projeto e Implementação	17
2.2	Desenvolvimento Web	18
2.2.1	O Ambiente Web	20
2.2.2	MVC - Padrão de Arquitetura de Software	21
2.3	O método FrameWeb	22
3	ESPECIFICAÇÃO DE REQUISITOS	24
3.1	Descrição do Escopo do Projeto	24
3.1.1	Cálculo de Pontuação	25
3.1.2	Fluxo de Trabalho no C2D	25
3.1.3	Atualização de Dados	26
3.2	Análise de Objetivos	27
3.3	Diagrama de Casos de Uso	28
3.4	Diagrama de Classes	29
4	PROJETO DE ARQUITETURA E IMPLEMENTAÇÃO	31
4.1	Arquitetura do Sistema	31
4.1.1	Camada de Apresentação	32
4.1.2	Camada de Negócio	33
4.1.3	Camada de Acesso a Dados	34
4.2	Modelos FrameWeb	34
4.2.1	Modelo de Entidades	34
4.2.2	Modelo de Persistência	35
4.2.3	Modelo de Navegação	35
4.2.4	Modelo de Aplicação	39
5	APRESENTAÇÃO	41

5.1	Login	41
5.2	Upload LattesCV	41
5.3	UploadQualis Data	43
5.4	Create New Score System	45
5.5	Calculate Academics Score	45
6	CONSIDERAÇÕES FINAIS	51
6.1	Conclusão	51
6.2	Limitações e Trabalhos Futuros	52
	REFERÊNCIAS	53
	APÊNDICES	55

1 Introdução

Anualmente, a CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) realiza avaliações de Programas de Pós-Graduação (PPGs) de acordo com diversos critérios, dentre eles sua produção bibliográfica, ou seja, publicação de artigos científicos em veículos (conferências e periódicos) qualificados. Assim, os PPGs elaboram critérios para que docentes sejam credenciados, se mantenham no programa, alcancem diferentes níveis dentro do programas etc., em torno das publicações dos professores e da qualificação que a CAPES faz dos veículos onde estes artigos são publicados, chamada de Qualis.

O Programa de Pós-Graduação em Informática da Universidade Federal do Espírito Santo (PPGI/UFES) conta com o Marvin,¹ um protótipo de sistema de informação Web que visa oferecer várias ferramentas úteis para atividades de ensino e pesquisa em uma universidade. Vários estudantes do Departamento de Informática da Universidade Federal do Espírito Santo (DI/UFES) desenvolvem ferramentas como projeto de graduação. O Marvin é um meio de integrar tais ferramentas de forma que viabilize seu uso e otimize os diversos processos dentro da universidade.

A partir disso e, considerando inicialmente os critérios estabelecidos pelo PPGI/UFES, surgiu a necessidade de um módulo que disponha de funcionalidades tal qual seja capaz de automatizar o processo de credenciamento e classificação dos docentes no programa de pós-graduação. Dessa forma, o módulo C2D (Classificação e Credenciamento de Docentes) surge com o intuito de facilitar a realização de atividades relacionadas ao credenciamento e gerenciamento de docentes do PPGI/UFES, controlando os principais dados para entrar no programa como docente.

1.1 Objetivos

O objetivo geral deste trabalho é desenvolver um sistema Web que será utilizado pelo PPGI/UFES a fim de auxiliá-lo em suas atividades de credenciamento e reconhecimento de docentes. Para isso, serão utilizados conceitos aprendidos ao longo do curso de Ciência da Computação. São objetivos específicos deste projeto:

- Realizar levantamento de requisitos e analisar objetivos do sistema de forma a apoiar esta fase;
- Produzir o documento de especificação e análise de requisitos do software;

¹ <<http://dev.nemo.inf.ufes.br:8180/Marvin>>.

- Produzir o documento de projeto de software, utilizando o método de desenvolvimento de software FrameWeb ([SOUZA, 2007](#));
- Implementar um protótipo do sistema na forma de um módulo a ser integrado ao Marvin.

Assim, tem-se o objetivo de desenvolver o sistema conforme definições no processo de Engenharia de Software, utilizando *frameworks* e ferramentas já existentes para auxiliar no desenvolvimento. Para esse objetivo, foram colocados em prática os conceitos de Programação, Linguagens de Programação, Banco de Dados, Desenvolvimento Web entre outros.

Vale ressaltar que foi realizado um trabalho de Iniciação Científica com a Prof^a. Renata S. S. Guizzardi, onde foram aprendidos e aplicados conceitos de Modelagem Conceitual, Engenharia de Requisitos e Análise de Objetivos.

Ao fim do projeto foram elencadas sugestões de futuras melhorias para suprir as limitações do sistema.

1.2 Metodologia

A metodologia utilizada no desenvolvimento deste trabalho consiste das seguintes atividades:

1. *Revisão Bibliográfica*: realizar um estudo das boas práticas de Engenharia de Software e de Engenharia de Requisitos, Modelagem Conceitual, Análise de Objetivos, Projeto de Sistemas, Programação Orientada a Objetos, Desenvolvimento de Software, Projeto de Banco de Dados, dentre outros;
2. *Desenvolvimento da Documentação do Sistema*: definir documentos do sistema. Primeiramente, foi produzido o Documento de Especificação de Requisitos, fornecendo uma descrição geral do mundo do sistema, definição dos requisitos funcionais e não funcionais, bem como as regras de negócio e o seu respectivo modelo de objetivo. Além disso, neste documento está a apresentação do sistema e de seu módulo/subsistema, casos de uso, modelo estrutural e glossário do projeto. Por fim, foi produzido o Documento do Projeto, no qual a arquitetura do software e projeto são detalhados seguindo as diretrizes propostas no método FrameWeb;
3. *Uso de Tecnologias*: estudo das tecnologias utilizadas para o desenvolvimento do sistema, tais como a linguagem de programação Java, os componentes da plataforma Java EE, a biblioteca de componentes de interface com o usuário PrimeFaces, o banco de dados MySQL, dentre outras;

4. *Implementação e Testes*: implementação das funcionalidades do sistema, com realização de testes informais para validação de seu funcionamento.
5. *Redação da Monografia*: escrita desta monografia, feita em *LaTeX*² utilizando o editor *TexStudio*³ e o template *abnTeX*⁴ o qual atende aos requisitos das normas da ABNT (Associação Brasileira de Normas Técnicas) para elaboração de documentos técnicos e científicos brasileiros.

1.3 Organização do Texto

O restante desta monografia é organizada em cinco partes e contém, além da desta introdução, os seguintes capítulos:

- **Capítulo 2** – Referencial Teórico: apresenta uma revisão da literatura a respeito dos temas relevantes para o contexto deste trabalho, sendo estes: Engenharia de Software, Análise de Objetivos, Desenvolvimento Web e o método FrameWeb;
- **Capítulo 3** – Especificação de Requisitos: apresenta a especificação de requisitos do sistema, a descrição do minimundo, modelos de objetivos do sistema, diagramas de classes e seus casos de uso;
- **Capítulo 4** – Projeto Arquitetural do sistema: apresenta a arquitetura utilizada no desenvolvimento do sistema, bem como os diagramas de projeto prescritos pelo método FrameWeb;
- **Capítulo 5** – Apresentação do sistema: apresenta as principais partes da implementação do sistema, ilustradas por capturas de telas;
- **Capítulo 6** – Considerações Finais: apresenta a conclusão do trabalho, dificuldades encontradas, limitações e propostas de trabalhos futuros.
- **Apêndice**: Contém os documentos produzidos nos processos de Engenharia de Software (Documento de Requisitos e Documento de Projetos)

² <<http://www.latex-project.org/>>.

³ <<http://www.texstudio.org/>>.

⁴ <<http://www.abntex.net.br>>.

2 Referencial Teórico

Neste capítulo, são apresentados os conceitos teóricos fundamentais para o desenvolvimento do sistema C2D, em 3 seções. A Seção 2.1 aborda a Engenharia de Software, frisando os principais processos e conceitos utilizados. A Seção 2.2 apresenta os principais conceitos de desenvolvimento de sistemas na plataforma Web. A Seção 2.3 apresenta o método FrameWeb.

2.1 Engenharia de Software

O desenvolvimento de software, de fato, é uma atividade de suma importância para nossa geração. A utilização de computadores e outros dispositivos nas mais diversas áreas do conhecimento humano tem gerado uma crescente demanda por soluções computadorizadas. Buscando melhorar a qualidade dos produtos de software e aumentar a produtividade no processo de desenvolvimento, surgiu a Engenharia de Software (FALBO, 2014).

Segundo Pressman (2011), a Engenharia de Software proporciona várias ferramentas para o desenvolvimento de um software de qualidade por meio da especificação, desenvolvimento e manutenção destes sistemas de software. Tudo isso é feito por meio de tecnologias e práticas de gerência de projetos e outras disciplinas, visando organização, produtividade e qualidade nesse processo de desenvolvimento (FALBO, 2014). Sendo assim, podemos citar as seguintes atividades do processo de desenvolvimento de software: elicitação de requisitos (especificação e análise), projeto arquitetural, implementação, análise de riscos, definição de cronogramas, dentre outros.

Como principais atividades no processo de desenvolvimento de software, podemos citar a fase de engenharia de requisitos e a fase de projeto do sistema. Porém, vale ressaltar que, paralelo a estas etapas principais, são realizados diversos processos para auxílio na gerência do projeto, como organização de cronograma, delegação de responsabilidades no projeto, análise dos riscos, testes etc.

Nas seções seguintes, serão apresentadas as atividades referentes a este processo de desenvolvimento de software. Na Seção 2.1.1, será abordada a etapa de especificação e análise de requisitos. Na seção 2.1.2 será abordada a etapa de análise de objetivos e os conceitos necessários para tal atividade. Na Seção 2.1.3, será abordada a etapa de projeto e implementação, sendo levantados conceitos relevantes a fim de explicar e exemplificar de forma clara a importância dessas atividades no contexto do desenvolvimento de software.

2.1.1 Engenharia de Requisitos

A Engenharia de Requisitos é considerada uma das atividades mais importantes no desenvolvimento de software, pois é nela que são definidas as necessidades do cliente e os possíveis problemas e limitações do projeto. O desenvolvimento da especificação de requisitos de um software é reconhecido como uma das bases das funcionalidades de um sistema. Estes requisitos são determinantes da qualidade do software, dado que estudos empíricos mostraram que erros nos requisitos são as falhas mais comuns no ciclo de vida de um software, bem como as mais caras e custosas a corrigir (AURUM et al., 2013).

Podemos definir um **requisito** como (ASSOCIATION et al., 1990):

1. Uma condição a ser alcançada por um sistema ou componentes de um sistema a fim de satisfazer um contrato, especificação ou outros documentos de formalização;
2. Um trabalho necessário para resolver um problema ou alcançar um objetivo;
3. Uma representação documentada de uma condição ou capacidade, de acordo com (1) e (2) anteriormente descritos;

A atividade inicia com o levantamento de requisitos, onde são coletadas as necessidades dos usuários, informações de domínio, restrições, sistemas interligados, regulamentos etc. A criação de modelos segue em paralelo à atividade de análise, onde estes representam o que o sistema faz. Esta atividade é conhecida como Modelagem Conceitual. Nesta etapa, o foco é o domínio do problema e não a solução técnica, linguagem a ser usada na implementação etc. Com essas informações, já se pode avançar no processo de desenvolvimento de software (FALBO, 2017).

A Figura 1 representa o ciclo de vida do processo de Engenharia de Requisitos, identificando suas principais atividades e dependências.

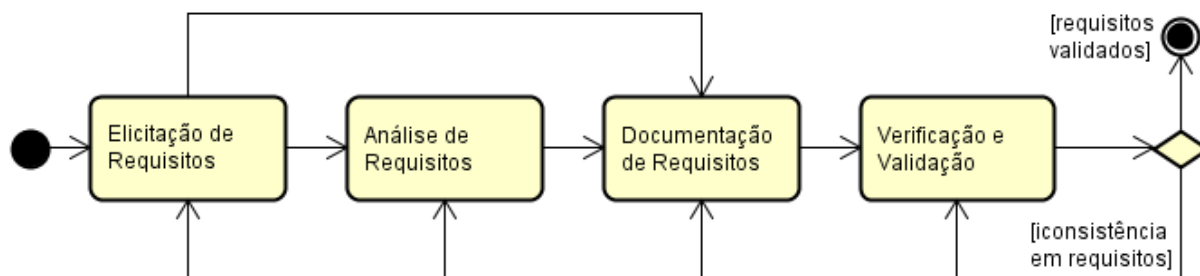


Figura 1 – Processo de Engenharia de Requisitos adaptado de (KOTONYA; SOMMERVILLE, 1998).

Assim, a especificação dos requisitos e análise das necessidades para o desenvolvimento de um software agrega muitas outras tarefas além de apenas perguntar aos *stakeholders* quais são suas necessidades. Requer uma profunda análise de todo o contexto

da organização e do domínio da aplicação com seus processos e regras de negócio.

Para melhor entendimento e visualização, são criados modelos, os quais descrevem as funcionalidades do software, deixando de lado a questão da implementação. Esta fase é chamada de Modelagem Conceitual, enfatizando o domínio do problema e não se deve pensar na solução técnica, computacional que será utilizada. Obtendo os requisitos, mesmo que de forma parcial, e especificando-os na forma de modelos, pode-se iniciar o trabalho no domínio da solução (FALBO, 2017).

Pode-se dizer que os requisitos de um sistema incluem especificações dos serviços que o sistema deve prover, restrições sob as quais ele deve operar, propriedades gerais do sistema e restrições que devem ser satisfeitas no seu processo de desenvolvimento (FALBO, 2017), sendo estes requisitos divididos em requisitos funcionais, requisitos não funcionais e regras de negócio, definidos segundo (AURUM et al., 2013) da seguinte forma:

- **Requisitos Funcionais:** o que o sistema efetivamente deve fazer;
- **Requisitos Não-funcionais:** especificação de um determinado comportamento do sistema. Por exemplo: usabilidade, interoperabilidade e segurança;
- **Regras de negócio:** usadas pela organização de forma a satisfazer os objetivos de negócio e clientes, segundo convenções definidas para o negócio.

2.1.2 Análise de Objetivos

A análise e modelagem de objetivos utilizando a linguagem iStar 2.0 (anteriormente i*) é uma tentativa de introduzir alguns aspectos de modelagem social e raciocínio em métodos de engenharia em sistemas de informação, especialmente no nível de requisitos. Diferente dos métodos tradicionais de análise de sistemas, em iStar tem-se a capacidade de reconhecimento de relações e intenções de atores no contexto social, focando em como as intenções dos atores (seja humano ou sistema) estão dispostas no domínio, podendo ser feitas alterações ou reconfigurações das relações de modo a ajudar nos objetivos dos atores (MYLOPOULOS; CHUNG; YU, 1999). Para isso, iStar apresenta alguns construtos como: *Actor*, *Agent*, *Role*, *Goal*, *Quality*, *Task*, *Resource* (DALPIAZ; FRANCH; HORKOFF, 2016). Alguns destes construtos são exemplificados na Figura 2 e explicados brevemente à seguir:

- *Actor* (Ator): entidade que possui objetivos e realiza ações para alcançá-los, exercendo o seu *know-how*;
- *Goal* (Objetivo): representa o desejo intencional de um ator. Os detalhes de como o objetivo é satisfeito não é descrito pelo objetivo. Portanto, pode ser descrito através de refinamento em tarefas;

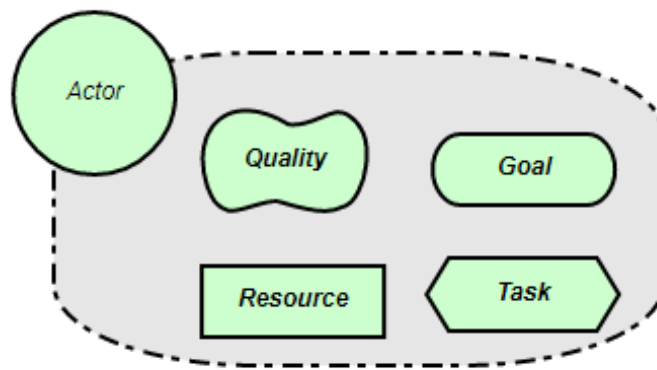


Figura 2 – Construtos da linguagem iStar 2.0.

- *Quality* (Qualidade): é um objetivo que se diferencia do *goal* por seu critério de satisfação, que é subjetivo. Um *quality* é considerado suficientemente satisfeito do ponto de vista do ator;
- *Task* (Tarefa): uma ação ou um conjunto de ações (passo a passo) realizadas pelo ator. O detalhamento da tarefa pode ser obtido pela decomposição de tarefa em outros subelementos;
- *Resource* (Recurso): entidade física ou informacional usada pelo ator. Assume-se que o recurso está disponível ao ator que o utiliza.

Além disso, existem links para representação de relações, descritos à seguir:

- *OR-Refinement link*: aponta para uma relação entre um fim e um meio que atinge esse fim. Um meio pode, por exemplo, ser uma tarefa ou um recurso e um fim pode ser um objetivo;
- *AND-Refinement link*: um objetivo pode ser decomposto em subobjetivos e uma tarefa pode ser decomposta em quatro tipos de elementos: um subobjetivo, uma subtarefa, um recurso, e/ou um *quality*;
- *Contribution link*: contribuições podem ser usadas para ligar qualquer um dos elementos a um *quality*, para modelar a maneira que o elemento afeta a satisfação ou cumprimento do *quality*;
 - Make (++): a contribuição é positiva o suficiente para satisfazer o *quality*;
 - Help (+): uma contribuição positiva parcial, não é suficiente por si só para satisfazer o *quality*;
 - Hurt (–): uma contribuição negativa parcial, não é suficiente por si só para negar o *quality*;

- Break (—): a contribuição é negativa o suficiente para negar o *quality*.

Há uma ferramenta online, disponível em <http://www.cin.ufpe.br/~jhcp/pistar/>, que permite a criação de diagramas usando a linguagem iStar 2.0.

Em suma, a análise de objetivos consiste em prover a captura das atividades e intenções de determinada organização ou sistema, por exemplo. Tal análise fortalece o alinhamento entre os recursos, requisitos necessários para desenvolvimento do sistema e regras, limitações também existentes, proporcionando uma visão geral do domínio e auxiliando nas decisões a serem tomadas para se chegar ao objetivo final, quer seja aprimorar estratégias em uma organização, quer seja o desenvolvimento de um sistema de forma eficaz.

2.1.3 Projeto e Implementação

Nesta fase, tem-se por objetivo definir o software a ser implementado. É na fase de projeto que se decide como a solução será implementada, visto que um mesmo problema pode ser resolvido de inúmeras formas possíveis. Desta forma, define-se o projeto da arquitetura do sistema, onde é descrita a estrutura de nível mais alto da aplicação, identificando seus elementos ou componentes e suas dependências entre si. A partir daí, é realizado um detalhamento desses elementos, até chegar no nível de unidades de implementação (FALBO, 2016), o qual corresponde à primeira atividade, focando nos aspectos tecnológicos, sendo, assim, a fase do processo de software na qual os requisitos, as necessidades do negócio e as considerações técnicas se juntam na formulação de um produto ou sistema de software (PRESSMAN, 2011).

Inicialmente, o projeto é representado em um nível alto de abstração, focando a estrutura geral do sistema. Definida a arquitetura, o projeto passa a tratar do detalhamento de seus elementos. Esses refinamentos conduzem a representações de menores níveis de abstração, até se chegar ao projeto de algoritmos e estruturas de dados. Assim, independentemente do paradigma adotado, o processo de projeto envolve as seguintes atividades (FALBO, 2016):

- **Projeto da Arquitetura do Software:** visa definir os elementos estruturais do software e seus relacionamentos.
- **Projeto dos Elementos da Arquitetura:** visa projetar em um maior nível de detalhes cada um dos elementos estruturais definidos na arquitetura, o que envolve a decomposição de módulos em outros módulos menores.
- **Projeto Detalhado:** tem por objetivo refinar e detalhar os elementos mais básicos da arquitetura do software, i.e., as interfaces, os procedimentos e as estruturas

de dados. Deve-se descrever como se dará a comunicação entre os elementos da arquitetura (interfaces internas), a comunicação do sistema em desenvolvimento com outros sistemas (interfaces externas) e com as pessoas que vão utilizá-lo (interface com o usuário), bem como se devem projetar detalhes de algoritmos e estruturas de dados.

Ao final da fase de Projeto e Implementação, espera-se que a arquitetura do sistema esteja definida, bem como o projeto de seus componentes, divididos geralmente em camadas a fim de assegurar a separação de funcionalidades pertencentes a nichos diferentes. Embora não seja a única, uma das estruturas mais conhecidas para o projeto de software é a que o divide em camada de apresentação, camada de domínio e camada de persistência (FOWLER, 2002). No contexto da Web, na camada de **apresentação**, espera-se lidar com requisições HTTP, exibição de informação e iterações do usuário (cliques de mouse, por exemplo). Na camada de **domínio**, como o próprio nome já diz, o foco está nas variáveis do domínio do sistema, implementando todas as regras de negócio e requisitos do sistema. Já na camada de **persistência**, espera-se realizar todo o relacionamento da aplicação com o banco de dados, permitindo armazenar as informações do sistema e recuperá-las quando necessário.

2.2 Desenvolvimento Web

A tecnologia está fortemente presente em nosso cotidiano. Um bom exemplo disso são as aplicações Web, executadas em computadores, tablets, smartphones etc, as quais podem ser utilizadas por quaisquer pessoas com acesso à Internet. Com isso, torna-se importante o processo de desenvolvimento de aplicações Web com características como manutenibilidade e escalabilidade.

Pode-se destacar diferenças entre o desenvolvimento de softwares voltados para Web e desenvolvimento de softwares tradicionais (Desktop): são sistemas cujo conteúdo está em constante alteração, buscando ter dados sempre mais atualizados e concisos, além de um rápido crescimento dos requisitos de sistemas na Web. Portanto, tais aplicações precisam ser cuidadosamente planejadas para suportar escalabilidade e permitir fácil manutenção, além de serem desenvolvidas com enfoque no usuário e possuir um ciclo de vida ágil e uma grande preocupação com acessibilidade, porquanto tais recursos não podem ser adicionados posteriormente ou necessitam de um esforço evitável. O sucesso na construção, desenvolvimento, implementação e manutenção de um sistema Web depende fortemente de quão bem essas questões foram tratadas.

Com o avanço da Web, surgiu a chamada Engenharia Web, que lida com o processo de desenvolvimento de aplicações e sistemas Web. Sua essência é de gerenciar a diversidade e a complexidade dessas aplicações, evitando potenciais falhas que possam ocasionar

implicações sérias. Pode-se dizer que a Engenharia Web é uma abordagem proativa de desenvolver aplicações Web (GINIGE; MURUGESAN, 2001).

A Java EE (Java Platform, Enterprise Edition) é uma plataforma padrão para desenvolver aplicações Java robustas e/ou para a Web, incluindo bibliotecas e funcionalidades para implementar softwares Java distribuídos. Se baseia em componentes modulares que executam em servidores de aplicações e que suportam escalabilidade, segurança, integridade e outros requisitos de aplicações corporativas (FARIA, 2013).

A plataforma Java EE possui inúmeras tecnologias com diferentes objetivos, por isso é considerada uma plataforma guarda-chuva. As especificações Java EE mais conhecidas são listadas em (MANZOLI, 2016):

- **Servlets:** são componentes Java executados no servidor para gerar conteúdo dinâmico para a Web, como HTML e XML, por exemplo;
- **JSF (JavaServer Faces):** é um *framework* Web baseado em Java com o objetivo de simplificar o desenvolvimento de interfaces de sistemas para a Web, por meio de um modelo de componentes as quais podem ser reutilizadas;
- **JPA (Java Persistence API):** é uma API padrão do Java para persistência de dados, baseada no conceito de mapeamento objeto/relacional. Essa tecnologia aumenta a produtividade para o desenvolvimento de sistemas onde a integração com banco de dados é necessária;
- **EJB (Enterprise Java Beans):** são componentes que executam em servidores de aplicação e possuem como principais objetivos fornecer facilidade e produtividade no desenvolvimento de componentes distribuídos, transacionados, seguros e portáteis;
- **CDI (Contexts and Dependency Injection):** especificação da Java EE que trabalha com injeção de dependências.

O Marvin (e, por conseguinte, o módulo C2D) também utiliza uma biblioteca de componentes de interface com o usuário chamada *Primefaces*.¹ O uso desta biblioteca otimiza o tempo gasto no desenvolvimento de componentes web responsivos (tabelas, campos de formulários, botões, dentre outros). Assim, podem ser criadas interfaces mais robustas e agradáveis ao olhar do usuário.

¹ <<http://www.primefaces.org/>>

2.2.1 O Ambiente Web

Para que as aplicações Web sejam executadas corretamente no computador do usuário, existem diversas tecnologias envolvidas neste processo e que se mostram de suma importância para a total compreensão do problema. Desde o surgimento da Web, o projeto evoluiu e são várias as razões para o seu atual sucesso, mas duas merecem destaque: sua arquitetura simples (mas eficiente) e uma interface igualmente simples, originalmente baseada no paradigma de hipertextos.

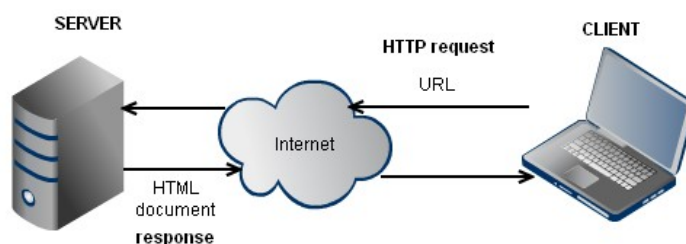


Figura 3 – Representação da Arquitetura de Rede *Cliente-Servidor*

Baseado na arquitetura cliente/servidor, o ambiente Web é formado por uma rede de computadores, onde cliente(s) e servidor(es) se comunicam através de requisições e solicitações, como mostra a Figura 3. Utilizando um *browser* (navegador), o cliente faz as requisições ao servidor — estas sendo apresentadas para o usuário da aplicação. Já no lado do servidor são recebidas e atendidas as requisições dos clientes. Os principais componentes desta arquitetura são o *HyperText Transfer Protocol* (HTTP): protocolo de comunicação utilizado para realizar as requisições por parte do usuário e resposta por parte do servidor; *Uniform Resource Locator* (URL): sistema de endereçamento o qual faz referência a um recurso na Internet; *HyperText Markup Language* (HTML): linguagem de marcação utilizada no desenvolvimento de páginas Web, permitindo criação de documentos que podem ser lidos por qualquer computador;

O **protocolo HTTP** é por onde os arquivos da Web são transmitidos, executado sobre a camada TCP/IP da Internet. Este protocolo pode ser organizado nos seguintes estados:

1. **Conexão:** O cliente estabelece uma conexão com o servidor;
2. **Requisição:** O cliente envia um pedido ao servidor;
3. **Resposta:** O servidor devolve uma resposta ao pedido do cliente;
4. **Encerramento:** A conexão é finalizada tanto pelo cliente quanto pelo servidor.

Quando um documento ou um objeto (como um arquivo de áudio, por exemplo) é

enviado para o cliente, é anexado um cabeçalho com a informação necessária para que o *browser* possa interpretá-lo e apresentá-lo ao usuário de forma adequada.

URL é a forma de referenciar objetos na Web. Consiste na identificação do esquema utilizado (HTTP, FTP, dentre outros) junto do caminho até o objeto ou documento desejado, como, por exemplo: `<http://aluno.ufes.br/>`.

Por fim, a linguagem **HTML** especifica a estrutura e a formatação para documentos do tipo hipertexto por meio de *tags* que indicam como estes devem ser visualizados.

2.2.2 MVC - Padrão de Arquitetura de Software

Com o advento da Internet e da procura cada vez maior por sistemas Web, fez-se necessário criar formas de agilizar o desenvolvimento destes sem comprometer a qualidade do produto final. Com isso, surgiu uma série de padrões de arquitetura de software em busca de atributos de qualidade como: (1) descentralização do desenvolvimento por dividir o código da aplicação em vários componentes, possibilitando que os desenvolvedores possam trabalhar paralelamente em diferentes componentes sem que isso cause impactos no que foi desenvolvido por outra pessoa; e (2) fraco acoplamento, uma vez que um dos princípios destes padrões de arquitetura é que suas camadas sejam o mais isoladas possível a fim de que umas não interfiram nas funcionalidades das outras.

Dentre os diversos padrões existentes, um dos mais utilizados é o padrão MVC: *Model-View-Controller*. Este padrão se baseia na divisão da aplicação em três camadas, conforme mostra a Figura 4. Neste modelo, cada camada possui uma funcionalidade específica:

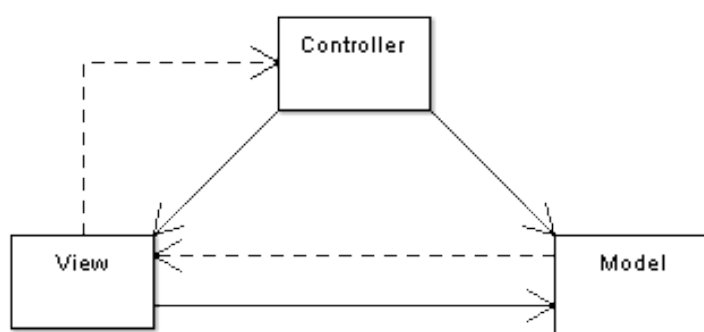


Figura 4 – Representação da Arquitetura de Software *Model-View-Controller*

- **Model:** camada que contém a lógica da aplicação, responsável pelas regras de negócio e pela comunicação da aplicação com o banco de dados;
- **View:** de forma sucinta, é a camada que interage diretamente com o usuário, exibindo dados por meio de arquivos HTML, XML, XHTML, PDF, entre outros;

- **Controller:** camada intermediadora do sistema, comunicando-se com as outras duas camadas. As requisições provenientes do *browser* são processadas pelo *controller*, o qual acessa as informações dos *models* e as retorna para as *views*, exibindo de forma adequada ao usuário.

2.3 O método FrameWeb

FrameWeb é um método de projeto para construção de sistemas de informação Web (Web Information Systems – WIS's) baseado em frameworks. O método define alguns tipos de frameworks que serão usados para construção da aplicação, definindo uma arquitetura básica para o WIS e propõe modelos de projeto que se aproximam da implementação do sistema usando esses *frameworks* (SOUZA, 2007).

De modo a se integrar bem com os *frameworks* utilizados, o FrameWeb define uma arquitetura lógica padrão para WISs baseada no padrão arquitetônico *Service Layer* (Camada de Serviço) (FOWLER, 2002). O sistema é dividido em três grandes camadas (SALVATORE, 2016):

- **Camada de Negócio (*Business Tier*):** responsável pelas funcionalidades relacionadas às regras de negócio da aplicação. Esta camada é particionada em duas: Lógica de Domínio (*Domain*) e Lógica de Aplicação (*Application*);
- **Camada de Apresentação (*Presentation Tier*):** responsável por funcionalidades de interface com o usuário (incluindo as telas que serão apresentadas a ele). Esta camada, segundo o padrão proposto, é também particionada em duas outras: Visão (*View*) e Controle (*Control*);
- **Camada de Acesso a Dados (*Data Access Tier*):** responsável por funcionalidades relacionadas à persistência de dados.

Além da definição desta arquitetura padrão, FrameWeb propõe um conjunto de modelos de projeto que trazem conceitos utilizados pelos *frameworks* para esta fase do processo por meio da definição de uma linguagem específica de domínio que faz com que os diagramas fiquem mais próximos da implementação (SOUZA, 2007; MARTINS, 2016).

Para representar componentes típicos da plataforma Web e dos *frameworks* utilizados, o FrameWeb estende o metamodelo da UML, especificando, assim, uma sintaxe própria. Com isso, é possível utilizá-lo para a construção de diagramas de quatro tipos:

- **Modelo de Entidades (*Entity Model*):** é um diagrama de classes da UML que representa os objetos de domínio do problema e seu mapeamento para a persistência em banco de dados relacional;

- **Modelo de Persistência** (*Persistence Model*): é um diagrama de classes da UML que representa as classes de acesso a dados (DAO) existentes, responsáveis pela persistência das instâncias das classes de domínio;
- **Modelo de Navegação** (*Navigation Model*): é um diagrama de classe da UML que representa os diferentes componentes que formam a camada de Lógica de Apresentação, como páginas Web, formulários HTML e *controllers*;
- **Modelo de Aplicação** (*Application Model*): é um diagrama de classes da UML que representa as classes de serviço, que são responsáveis pela codificação dos casos de uso, e suas dependências;

3 Especificação de Requisitos

Neste capítulo são abordados os resultados obtidos na fase de Engenharia de Requisitos para a construção do sistema de Classificação e Credenciamento de Docentes (C2D). Na Seção 3.1, é apresentado o minimundo do projeto; na Seção 3.2 são analisados os objetivos da aplicação a ser construída; na Seção 3.3, são apresentados diagramas de casos de uso e na Seção 3.4, são apresentados os diagramas de classes. Os requisitos funcionais, requisitos não funcionais e regras de negócio estão localizados no Documento de Especificação de Requisitos, disponível no Apêndice ao final desta monografia.

3.1 Descrição do Escopo do Projeto

Programas de Pós-Graduação (PPGs) são avaliados pela CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) de acordo com sua produção técnica, ou seja, publicação de artigos científicos em veículos (conferências e periódicos) qualificados. Por esta razão, os PPGs elaboram critérios para que docentes sejam credenciados no programa, se mantenham dentro do programa, alcancem diferentes níveis dentro do programa, etc. Todos estes critérios giram em torno das publicações dos professores e da qualificação que a CAPES faz dos veículos onde estes artigos são publicados, chamada de Qualis.

Além destas informações, foram usados documentos de requisitos iniciais já desenvolvidos anteriormente em outros projetos de graduação e iniciação científica, disponibilizados pelos alunos envolvidos, os *stakeholders* do projeto. Feita a revisão de tais documentos, foram extraídas as informações pertinentes e alterações foram feitas conforme o escopo deste projeto.

O C2D deve considerar, inicialmente, os critérios estabelecidos pelo PPGI/UFES, mas sendo também genérico o suficiente para permitir atualização destes critérios (caso mudem no futuro), podendo, assim, também ser aproveitado por outros PPGs. Para efeito de Minimundo, no entanto, consideram-se os critérios de publicação atuais do PPGI.

O PPGI define atualmente 4 critérios: (a) credenciamento (ingresso) no programa; (b) recredenciamento (manutenção) no programa; (c) categorização como colaborador ou permanente; e (d) habilitação para orientação no doutorado. Tais critérios envolvem vários aspectos, como: ter plano de trabalho aprovado, ter lecionado disciplinas, ter concluído orientações, ter vínculo funcional-administrativo e, por fim, atender a requisitos mínimos de publicação.

A princípio, o **C2D deve avaliar apenas os requisitos de publicação**. Neste contexto, os critérios (a) e (b) utilizam um requisito de publicação, enquanto o critério (d)

utiliza um outro requisito de publicação. Em 2017, o requisito para ingresso/permanência no programa é somar 20 pontos de publicação no último biênio, enquanto o requisito para orientar no doutorado é somar 25 pontos no biênio, com 10 pontos advindos de publicações em periódicos no triênio.

3.1.1 Cálculo de Pontuação

Tal pontuação é calculada de acordo com a classificação do veículo de publicação (conferência ou periódico) no Qualis da CAPES. Atualmente, a pontuação é dada seguindo as regras abaixo:¹

- Publicações em periódicos com Qualis B1 ou superior acumulam 20 pontos;
- Publicações em periódicos com Qualis B2 ou B3 acumulam 10 pontos;
- Publicações em periódicos com Qualis B4 ou B5 acumulam 2 pontos;
- Publicações em conferências com Qualis B1 ou superior acumulam 10 pontos;
- Publicações em conferências com Qualis B2 ou B3 acumulam 5 pontos.

Para computação dos pontos será utilizado o Qualis do ano da publicação ou o mais recente divulgado (para publicações sem Qualis deve-se usar os índices JCR² e *h-index*³ com parametrização equivalente aos extratos Qualis correspondentes). Desta forma, o critério de credenciamento fica independente das mudanças que podem ocorrer no Qualis. Adicionalmente, o critério é aplicável a qualquer periódico e evento, independentemente de constar nas listas do Qualis. Mais detalhes podem ser vistos no site do PPGI: <<http://www.informatica.ufes.br/pt-br/pos-graduacao/PPGI/credenciamento-de-docentes>>.

3.1.2 Fluxo de Trabalho no C2D

Considerando que dados sobre membros do PPGI, suas publicações, respectivos fatores de impacto e classificação Qualis, etc. encontram-se já registrados em outros sistemas e documentos, o C2D deve funcionar de maneira mais automatizada possível, obtendo estes dados e armazenando em sua base de dados local, atualizando-os sempre que necessário. Então, temos o seguinte fluxo de trabalho para o C2D:

1. Utilizando a função de cadastro presente no módulo núcleo (*core*) do Marvin, a secretária do PPG realiza seu cadastro no sistema, informando seu nome, e-mail, CPF

¹ <<http://www.informatica.ufes.br/pt-br/pos-graduacao/PPGI/credenciamento-de-docentes>>.

² <<http://www.periodicos.ufscar.br/noticias/o-que-e-o-journal-citation-reports-jcr>>

³ <https://pt.wikipedia.org/wiki/Índice_h>

e senha para que possa administrar o sistema. Após acesso ao sistema, a secretária pode efetuar também o cadastro dos docentes no sistema.

2. A secretária cadastra manualmente os diferentes níveis de classificação do Qualis (A1, A2, B1, B2, B3, etc.), o sistema de pontuação vigente e os requisitos de publicação vigentes, podendo cadastrar quantos requisitos quiser (atualmente o PPGI usa dois, como descrito anteriormente). Para cada requisito, além do nome e da vigência (data de início e fim), é possível associar a cada nível de classificação uma pontuação nacional e uma internacional;
3. A secretária pede ao C2D que extraia informações de classificação Qualis de conferências e periódicos. Atualmente, a classificação de conferências pode ser encontrada em uma tabela no documento PDF disponível no site da CAPES.⁴ Já a classificação de periódicos encontra-se na plataforma Sucupira,⁵ clicando na figura do Qualis e preenchendo o formulário de consulta. Note que nem o documento do Qualis Conferências, nem o sistema Sucupira indicam se o veículo é nacional ou internacional, portanto tal informação deve ser completada manualmente;
4. A última extração de dados que a secretária pode pedir ao C2D para fazer é recuperar as publicações dos docentes a partir de seus Currículos Lattes;
5. Com os docentes, critérios, qualificação e publicações cadastrados (atualizados), a secretária finalmente solicita ao C2D que calcule a pontuação de publicação de cada docente e informa, para cada um deles, se atendem aos requisitos cadastrados (no caso do PPGI, os dois requisitos que são avaliados atualmente).

3.1.3 Atualização de Dados

O cadastro de docentes (passo 2 acima) pode ser repetido de tempos em tempos, quando a secretária assim desejar. Esta funcionalidade se encontra no módulo núcleo (*core*) do Marvin, onde podem ser visualizados os docentes atualmente cadastrados, docentes a manter como estão (não muda nada), docentes que mudaram de tipo, novos docentes a cadastrar (presentes na lista de docentes do site mas não no cadastro) e docentes a remover do programa (presentes no cadastro mas não na lista de docentes).

O cadastro da classificação Qualis (passo 3) também deve ser repetido de tempos em tempos, fazendo uma atualização semi-automática similar à descrita para os docentes. É importante ressaltar que o veículo deve ser cadastrado uma só vez e ter associado a ele múltiplas classificações, cada uma com seu(s) ano(s) de vigência.

⁴ <http://capes.gov.br/images/documentos/Qualis_periodicos_2016/Qualis_conferencia_ccomp.pdf>

⁵ <<https://sucupira.capes.gov.br/sucupira/>>.

A extração de dados do Lattes (passo 4) é limitada pelo fato deste processo utilizar *captcha* para impedir a obtenção automática dos currículos. Neste caso, a solução utilizada emprega a extração manual dos dados: a secretária obtém o currículo em formato XML manualmente e faz o *upload* do mesmo no sistema. Além disso, os dados de publicação nos currículos são informados pelos próprios docentes e não há garantia que o nome dos veículos cadastrados por eles em seus currículos bate com o nome dos veículos obtidos na lista de conferências / periódicos do Qualis. O C2D deve tentar fazer o máximo possível para reconhecer automaticamente os veículos de cada publicação, utilizando a abordagem semi-automática sempre que necessário.

3.2 Análise de Objetivos

Com o escopo definido e o minimundo descrito, foi construído um modelo de objetivos para o módulo C2D do Marvin, com o intuito de mapear os principais objetivos e possíveis tarefas no sistema, auxiliando, assim, o processo de análise dos requisitos. A Figura 5 representa o modelo de objetivos, o qual foi criado utilizando a ferramenta online piStar⁶ para modelagem com a linguagem iStar 2.0.

Considerou-se o principal objetivo (*goal*) do sistema **Ter dados de docentes gerenciados**. Ligados a este objetivo existem outros objetivos secundários, os quais serão explicados a seguir.

Para o subobjetivo **Ter dados de venue gerenciados**, tem-se a tarefa (*task*) de classificar veículos de acordo com o recurso (*resource*) associado, que neste caso, é um Qualis. Outra tarefa associada a este objetivo é a criação de um sistema de pontuação, onde o recurso utilizado é um quadro de pontuação cujo conteúdo são os valores de cada Qualis.

Prosseguindo, para o subobjetivo **Ter dados de publicação gerenciados** tem-se a tarefa de classificar as publicações, de acordo com a sua categoria (conferência ou periódico) e com as informações contidas nos arquivos .csv relacionados.

Para o subobjetivo **Ter dados de pesquisador gerenciados**, é preciso ter o cadastro deste efetuado. No cadastro de docente, é necessário incluir o seu currículo Lattes. A outra tarefa é calcular sua pontuação. Quando o cálculo passa a ser feito por meio do sistema, o tempo gasto nesta tarefa é reduzido. Tais fatos são representados no modelo pelo construto *Quality*. O mesmo ocorre na autenticação do administrador. Esta tarefa faz com que o acesso ao sistema seja restrito, portanto mais seguro.

A relação entre a secretária e o C2D é representada pela tarefa de manipular todos os dados através do gerenciamento do sistema, como mostrado na Figura 5. O modelo

⁶ <<http://www.cin.ufpe.br/~jhcp/pistar/>>

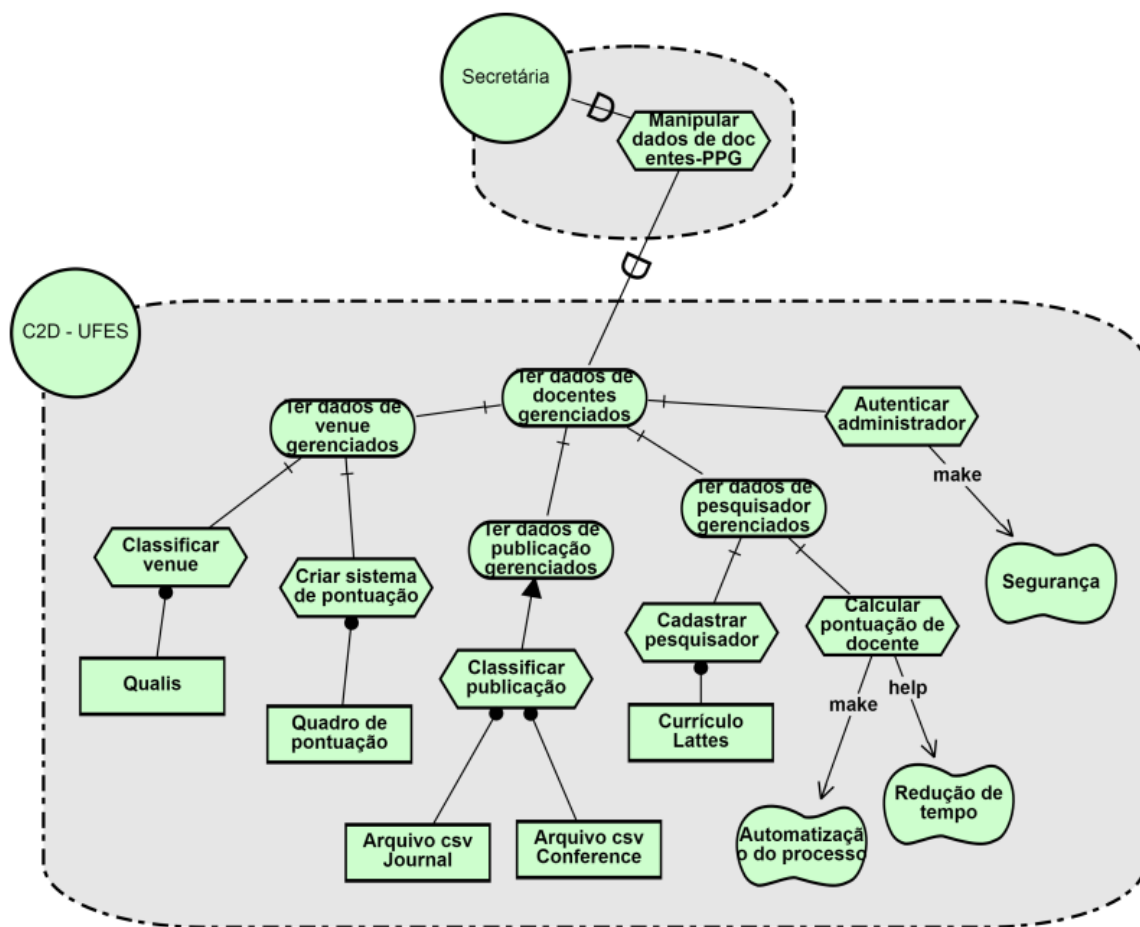


Figura 5 – Modelo de Objetivos do Módulo C2D.

auxilia na análise do sistema, mapeando as tarefas que devem ocorrer para que os objetivos e intenções dos atores sejam satisfeitos.

3.3 Diagrama de Casos de Uso

Nesta seção são descritos os casos de uso relacionados ao módulo C2D e atores envolvidos na aplicação. Apesar do Marvin envolver diferentes atores no contexto de uma universidade, o módulo C2D foca na secretária do PPG. A Figura 6 mostra o diagrama de casos de uso para o C2D.

Como descrito na Seção 3.1.2, a secretária realiza seu cadastro logo que o sistema é instalado, informando seu nome, e-mail, CPF e a uma senha de acesso. Após isso, é necessário cadastrar os docentes do PPG informando seus principais dados, como nome completo, CPF, e-mail, data de nascimento e link para currículo Lattes. Essas funcionalidades estão implementadas no módulo *core* (núcleo). No C2D, a secretária tem a tarefa de indicar quais são os docentes pertencentes ao PPG cadastrado, processo representado pelo caso de uso **Cadastrar PPG**.

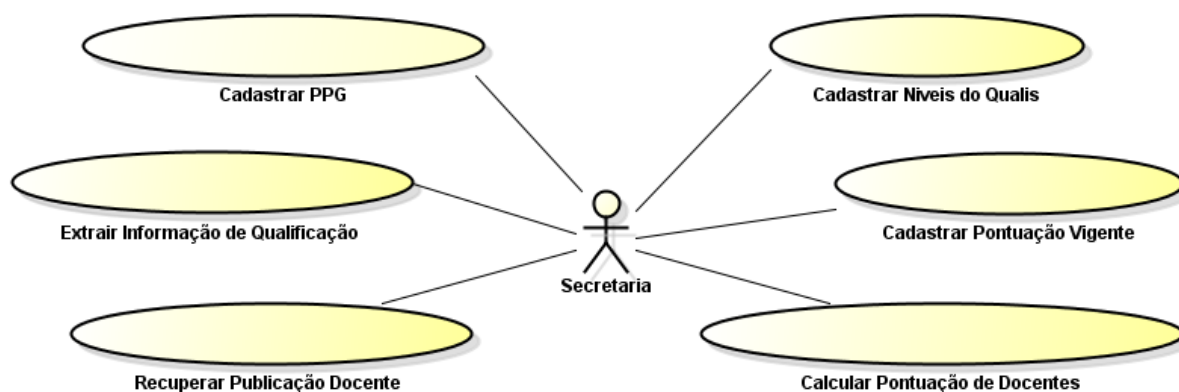


Figura 6 – Diagrama de Casos de Uso do Módulo C2D.

O caso de uso **Cadastrar Níveis de Qualis** representa o cadastro manual do Qualis feito pela secretária, enquanto **Cadastrar Pontuação Vigente** representa o processo de atualização de Qualis, pois seus valores podem ser alterados. Vale ressaltar que este processo se refere à pontuação vigente estabelecida pelo PPGI e não por meio do Qualis definido pela CAPES.

Assim, com o cadastro de docentes realizado e Qualis definidos, são descritos agora casos de uso envolvendo os dados de publicação: para as publicações, definidas como *Journal* ou *Conference*, existem arquivos (tabelas) com definições de pontuação e qualificação, de acordo com o veículo informado. Para o caso de uso **Extrair Informação de Qualificação**, o sistema C2D faz a leitura desses dados e os armazena para que possam ser usados. Para **Recuperar Publicação de Docente**, o sistema associa o código Lattes do currículo de um docente (fluxo descrito na Seção 3.1.2). Para o caso de uso **Calcular Pontuação de Docente**, a secretária informa o período e docente(s) que deseja calcular a pontuação, obtendo tal informação do sistema.

3.4 Diagrama de Classes

A Figura 7 apresenta o diagrama de classes do módulo C2D. As classes e suas associações são descritas a seguir.

As informações de docentes as quais a secretária irá administrar são associadas à classe **Academic**, pertencente ao módulo núcleo (*core*) do Marvin.

Um acadêmico (**Academic**) pode ser cadastrado sem possuir publicações associadas ao seu nome, ao passo que uma publicação (**Publication**) deve ser registrada com o nome do seu autor correspondente. Publicações são publicadas em um determinado veículo (**Venue**) e este, por sua vez, pode ter várias publicações associadas ou nenhuma associação. Para o sistema C2D, foi feita somente a distinção de veículos do tipo conferência (*Conference*) e periódico (*Journal*), armazenados na classe **VenueCategory**.

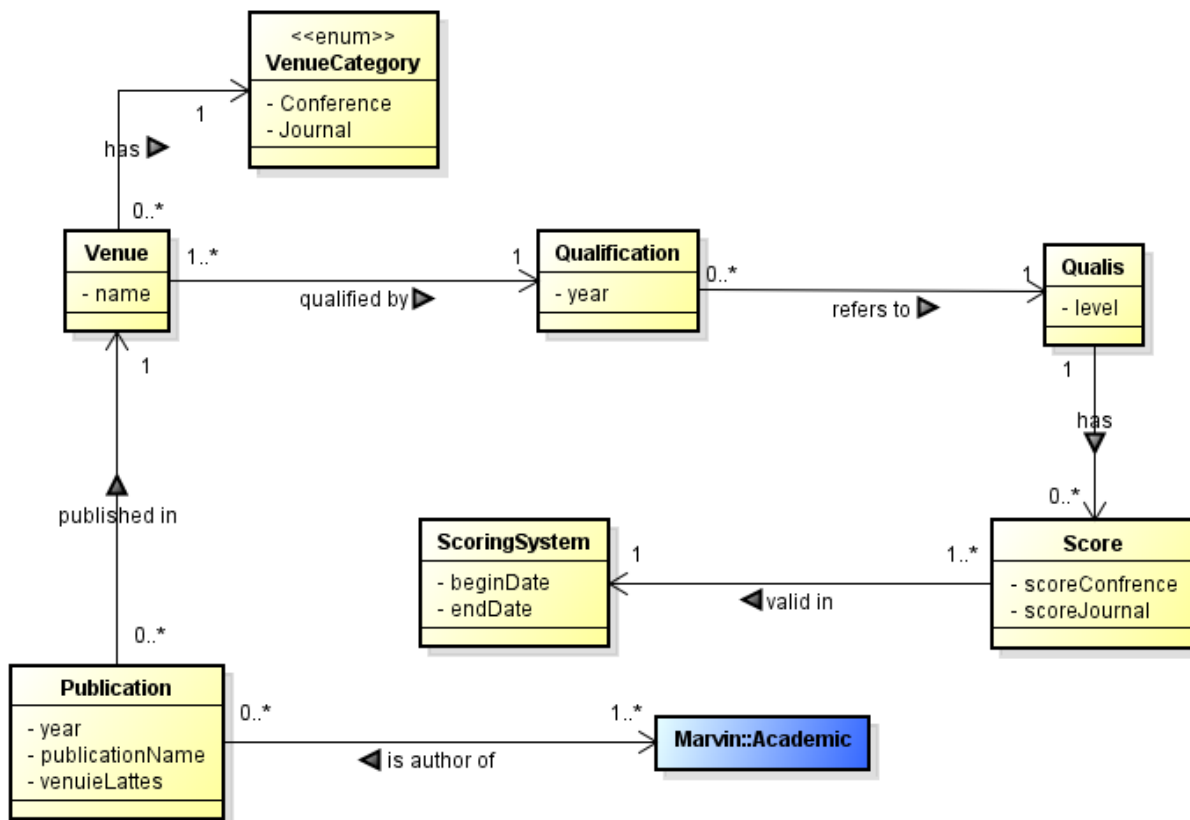


Figura 7 – Diagrama de Classes do Módulo C2D.

Cada veículo é também classificado por Qualis, referentes ao ano vigente, definido em Qualification. Qualis pode ser do tipo A1, A2, B1, B2, B3, B4, B5 e C. Estes itens são cadastrados com seus respectivos níveis. Contudo, o sistema permite que novos níveis de Qualis sejam cadastrados posteriormente.

Assim, a secretária pode solicitar ao C2D que extraia os dados de acadêmico necessários para o cálculo da pontuação segundo suas publicações e de acordo com um sistema de pontuação (ScoreSystem), definindo o período que se deseja calcular a pontuação (Score).

4 Projeto de Arquitetura e Implementação

Após as fases de especificação e análise de requisitos, realiza-se a fase de projeto do sistema. Esta etapa é responsável pela modelagem do sistema e de como ele será implementado, quais tecnologias serão utilizadas, de acordo com os requisitos levantados. Neste projeto foi utilizado o método FrameWeb (Seção 2.3) para o projeto da aplicação. Na Seção 4.1 é descrita a arquitetura do sistema, enquanto na Seção 4.2 são apresentados os modelos FrameWeb.

4.1 Arquitetura do Sistema

Inicialmente, o sistema Marvin era composto do subsistema *core* e suas respectivas funcionalidades. Criado o módulo C2D, precisou-se ser criado seu subsistema, segundo a fase de análise de requisitos descrita no Capítulo 3. O módulo `br.ufes.inf.nemo.marvin.core` contém as classes mais importantes: *Academic* (docente) e *Course* (curso) e seus respectivos cadastros. No módulo `br.ufes.inf.nemo.marvin.research` contém as funcionalidades do subsistema *C2D*.

Para os módulos existentes no sistema Marvin, foi utilizada a arquitetura proposta na Seção 2.2, apresentada na Figura 8 à seguir.

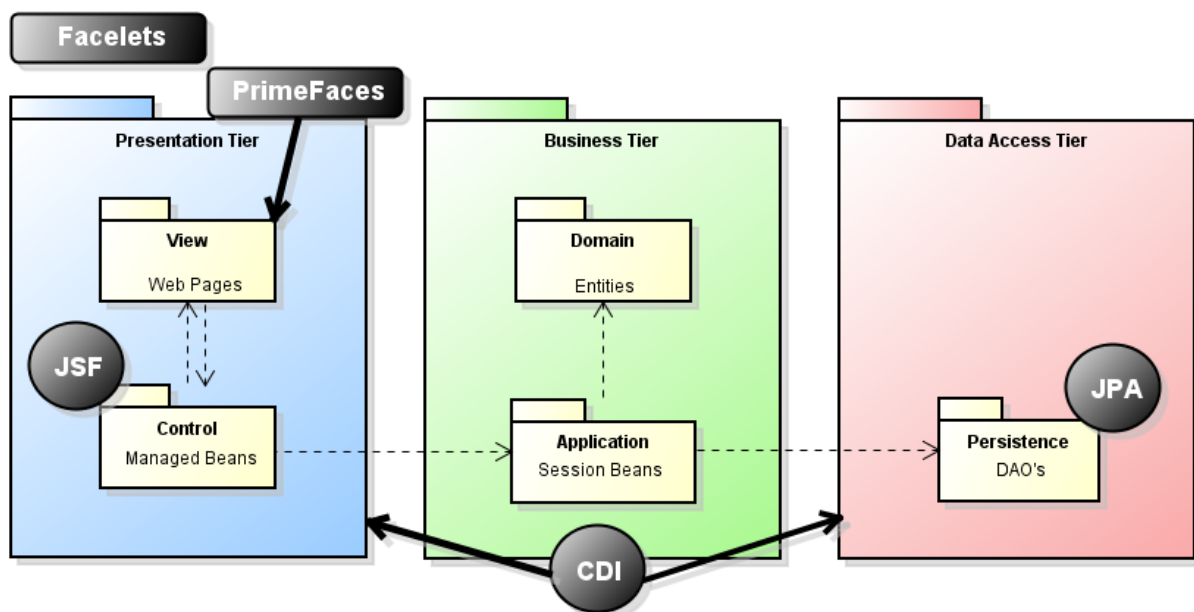


Figura 8 – Modelo de pacotes para o sistema Marvin (LIMA, 2015).

O módulo C2D é dividido em três camadas: camada de apresentação (*Presentation Tier*), camada de negócio (*Business Tier*) e camada de acesso a dados (*Data Access Tier*).

As associações dessas camadas com as tecnologias de Java EE anteriormente citadas na Seção 2.2 são também representadas na Figura 8.

Para o desenvolvimento das interfaces do módulo C2D, foram reutilizados componentes de interface já existentes no sistema Marvin, utilizando Facelets, reutilizando estruturas de páginas de visão da aplicação usados no *framework* JSF. Além disso, utiliza-se a biblioteca PrimeFaces, que possui diversos componentes para agilizar o processo de desenvolvimento de interfaces para sistemas Web.

O CDI é utilizado para fazer as ligações entre o pacote `Control` e o pacote `Application` e entre o pacote `Application` e o pacote `Persistence`. A função desta tecnologia é fazer com que um *Managed Bean* tenha acesso a um *Session Bean*, por exemplo. Para isso, utiliza-se anotações específicas como `@Inject`, `@PersistenceContext` e `@EJB`, por exemplo, de acordo com a dependência e escopo em que um Bean deve ser injetado, sem que seja necessário explicitar isso em termos de código Java.

O JPA é utilizado para fazer a persistências de dados. A integração com o banco de dados é feita também utilizando anotações, onde uma classe Java recebe a anotação `@Entity`. Tal anotação relaciona esta classe a uma tabela no banco de dados da aplicação. Já para atributos, pode-se usar anotações `@Basic`, `@Temporal` ou `@Transient`, de acordo com a forma de persistência que se deseja usar. Além disso, o JPA se encarrega de fazer as relações entre as classes do domínio, por meio das anotações `@OneToOne`, `@OneToMany`, `@ManyToOne` e `@ManyToMany`.

A Figura 9 mostra o sistema Marvin, dividido por módulos. Na figura o módulo C2D é apresentado como `br.ufes.inf.nemo.marvin.research`.¹

4.1.1 Camada de Apresentação

A camada de apresentação está dividida nos pacotes de visão e controle. No pacote de visão encontram-se as páginas Web do sistema Marvin, enquanto que no pacote de controle os controladores responsáveis pela comunicação entre as interfaces e aplicações do sistema. A Figura 10 representa a estrutura de páginas Web do módulo C2D.

Os arquivos relacionados à visão do sistema estão localizados na pasta `WebContent`. Dentro dela, existem as pastas `core` e `research`, representando os módulos do sistema. Já dentro de `research` foram criadas subpastas para salvar as páginas de acordo com os casos de uso do módulo C2D.

Dentro da pasta `WebContent` também estão as páginas `index.html` e `index.xhtml`, as páginas iniciais do sistema e uma pasta `resource/default`, que possui um decorador que é usado para definir o *layout* da página e o menu que está sendo utilizado.

¹ As tecnologias JPA, JSF e CDI foram apresentadas na seção 2.2

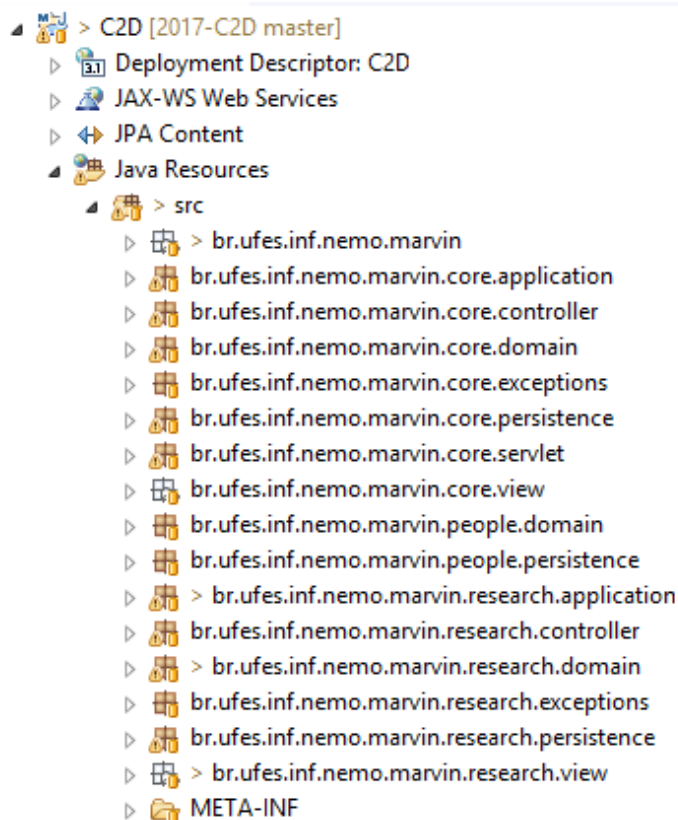


Figura 9 – Pacotes na Implementação do C2D.

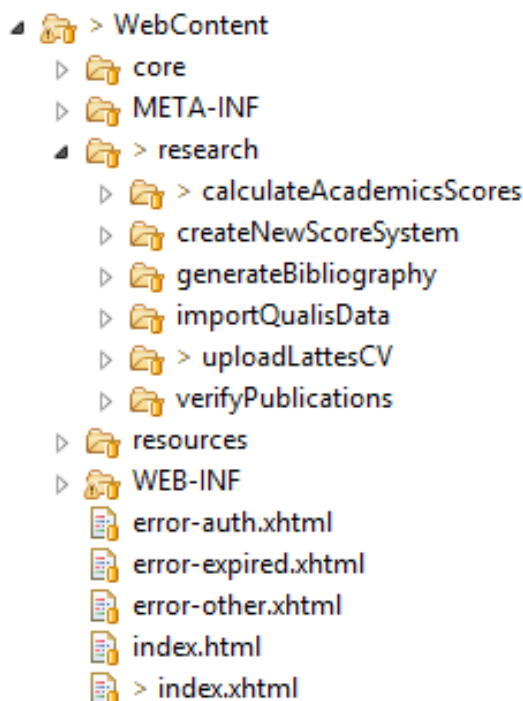


Figura 10 – Estrutura de páginas do sistema Marvin.

4.1.2 Camada de Negócio

A camada de negócio está dividida nos pacotes de domínio e aplicação. No pacote de domínio encontram-se as classes de domínio (entidades de negócio), enquanto que no

pacote de aplicação a implementação dos casos de uso e classes utilizadas para manipulação e validação de dados.

4.1.3 Camada de Acesso a Dados

A camada de acesso a dados é responsável pela persistência dos dados. Sua implementação está no pacote `br.ufes.inf.nemo.marvin.research.persistence`, contendo estes DAO's, objetos responsáveis pela comunicação com banco de dados, armazenamento e recuperação de dados.

4.2 Modelos FrameWeb

Nesta seção são apresentados os modelos FrameWeb, de acordo com a ordem na qual foram apresentados na Seção 2.3.

4.2.1 Modelo de Entidades

Primeiramente, temos o **Modelo de Entidades**, exibido na Figura 11. Os mapeamentos de persistência são metadados das classes de domínio, permitindo que *frameworks* ORM realizem o mapeamento objeto/relacional de forma automática, transformando os objetos que estão em memória em linhas de tabelas no banco de dados relacional.

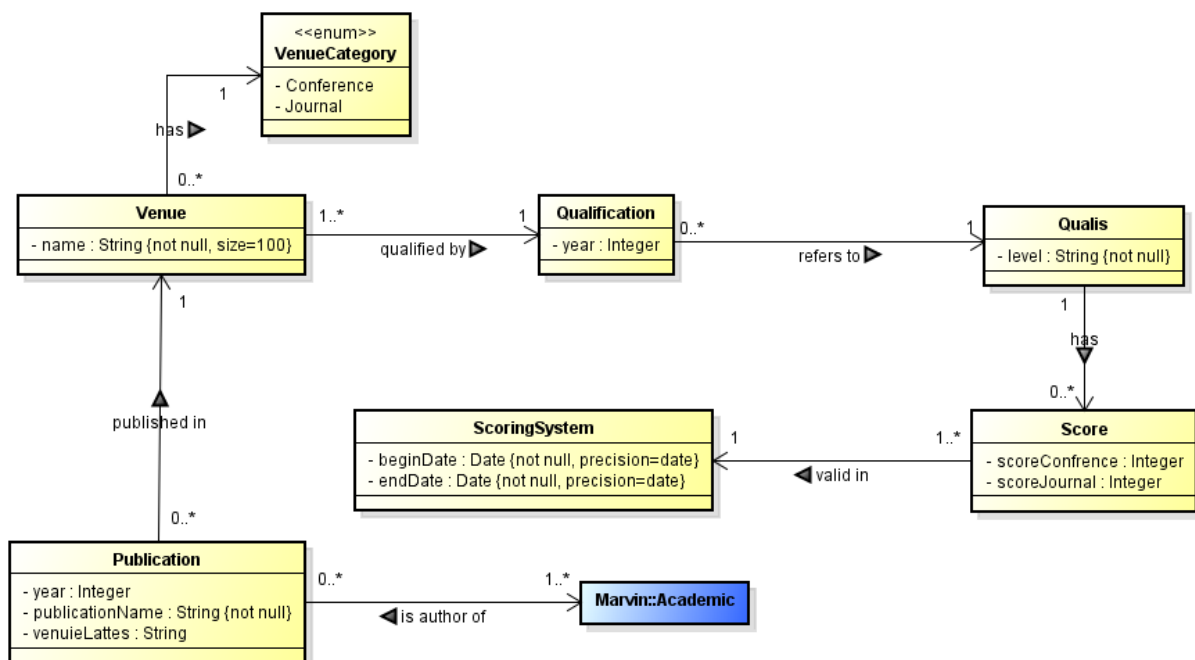


Figura 11 – Modelo de Entidades do módulo C2D.

Baseado no padrão do FrameWeb proposto em 2007 e no modelo de classes definido na fase de requisitos, a navegabilidade de cada associação é definida. Além disso, todos os atributos que são não nulos a tag *not null* também é exibida.

4.2.2 Modelo de Persistência

O **Modelo de Persistência** é um diagrama de classes da UML que representa as classes DAO existentes. O DAO - *Data Access Object* - é responsável pela persistência das instâncias das classes de domínio.

O modelo apresenta, para cada classe de domínio que precisa da lógica de acesso ao banco de dados, uma interface e uma classe concreta DAO que implementa a interface. Esta interface, única, define os métodos de persistência existentes para aquela classe, os quais serão implementados por uma ou mais classes concretas, de acordo com a tecnologia de persistência utilizada (ex.: DAO para o *framework* Hibernate) (SOUZA, 2007).

Para que não haja repetição de funcionalidades como salvar objetos, retornar todos os objetos, atualizar, etc., todas as classes DAO existentes herdam as propriedades da classe *BaseDAO*, apresentada na Figura 12.

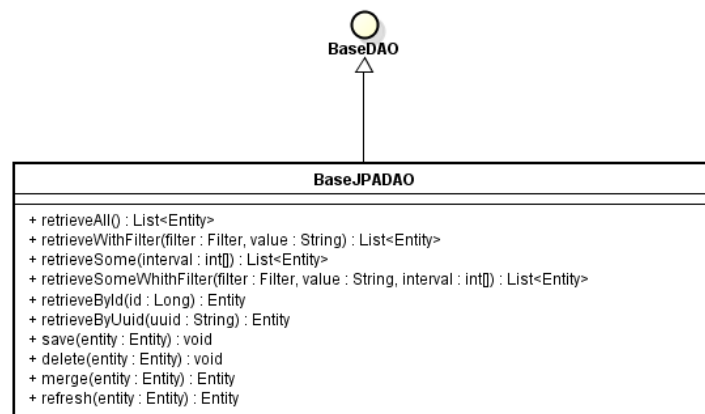


Figura 12 – Classe *BaseDAO* e principais métodos.

A Figura 13 apresenta as classes DAO's existentes no sistema C2D.

4.2.3 Modelo de Navegação

Por fim, tem-se o **Modelo de Navegação**, onde são apresentadas classes da UML, as quais representam os diversos componentes que formam a camada de Apresentação, como páginas Web, classes de controle e formulários HTML. Esse modelo é utilizado pelos desenvolvedores para orientar na codificação das classes e componentes das camadas *View* e *Control*. A classe de controle é o principal componente deste modelo: suas associações de dependência indicam o controle de fluxo quando uma ação é executada.

As funcionalidades de CRUD — *create, read, update, delete* — seguem um mesmo fluxo de execução e de interação com o usuário. Essas funcionalidades são bastante parecidas em sua essência para todos os casos de uso cadastrais, visto que foi utilizado Java EE para o desenvolvimento do projeto. Tal fluxo é apresentado na Figura 14, apresentado por Salvatore (2016) no desenvolvimento de um outro módulo do Marvin. Neste caso,

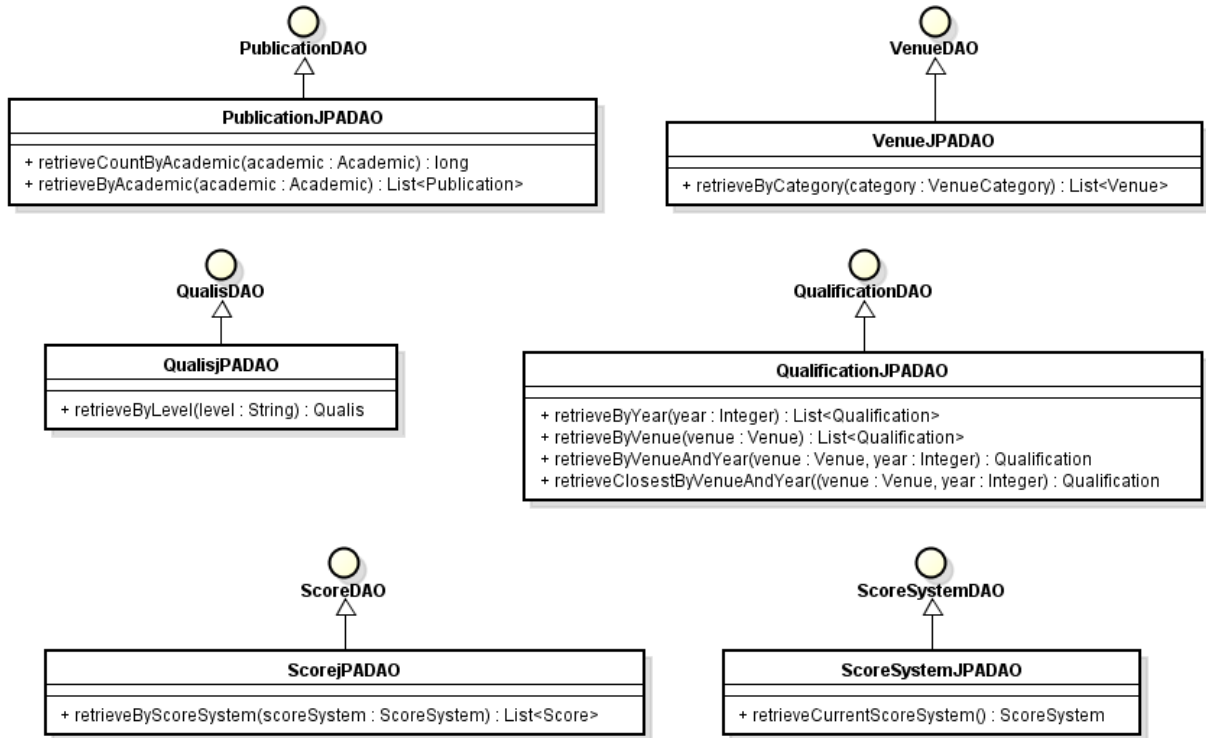


Figura 13 – Modelo de Persistência do módulo C2D.

utilizou-se de um modelo genérico como referência para modelos de navegação para casos de uso cadastrais, como Cadastrar Níveis de Qualis e Cadastrar Pontuação Vigente, utilizando o método *create* e os casos de uso Cadastrar Acadêmico e Cadastrar Secretária, do módulo núcleo (*core*) do sistema por exemplo.

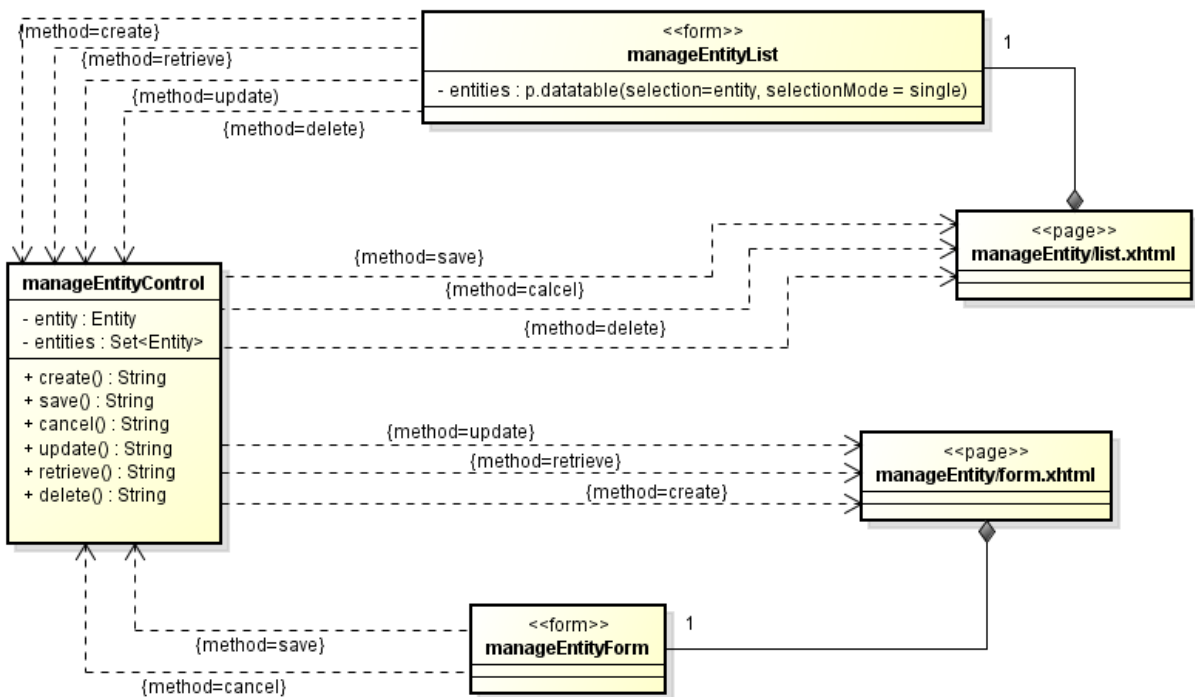


Figura 14 – Modelo de Navegação CRUD (SALVATORE, 2016).

A página `list.xhtml` possui uma listagem dos objetos de determinado tipo existentes no banco de dados. A exibição destas entidades é feita em seu formulário `manageEntityList` por meio de um componente tabela da biblioteca PrimeFaces (`p.dataTable`). Ao selecionar uma entidade da tabela, esta entidade é alocada no atributo `entity` de `ManageEntityControl`, possibilitando então executar ações como atualizar o objeto selecionado (`update`), ler o objeto selecionado (`retrieve`) ou excluir o objeto selecionado (`delete`). Independente de haver uma entidade selecionada na tabela, pode-se também criar um novo objeto (`create`). Essas ações, ao serem executadas, chamam os seus respectivos métodos do controlador. As relações de dependência mostradas na Figura 14 representam as formas de retorno para cada ação realizada. Por exemplo, caso se queira atualizar dados de alguma entidade, executa-se o método `update` e o usuário é redirecionado para a página `form.xhtml`, onde poderá fazer as alterações desejadas e, assim, executar o método `save` para salvar tais alterações. Após isso, usuário é redirecionado para a página inicial.

O modelo de navegação para o caso de uso **Calcular Pontuação de Docentes** é apresentado na Figura 15.

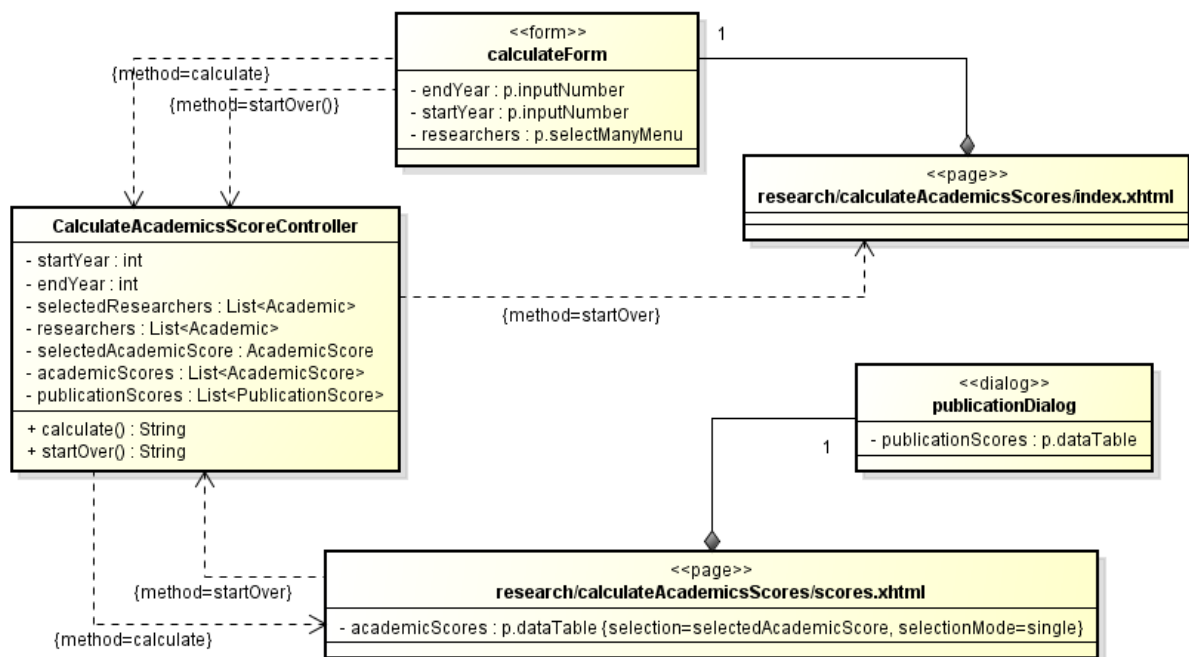


Figura 15 – Modelo de Navegação para fluxo de cálculo de pontuação.

A página `index.xhtml` possui o formulário `calculateForm` onde são preenchidos os campos de ano início (`startYear`), ano fim (`endYear`) e o acadêmicos que deseja calcular a pontuação (`selectedResearchers`). Já a página `scores.xhtml` representa a página onde são apresentados os resultados do cálculo da pontuação dos docentes selecionados. Esta página contém um `dialog`, o qual apresenta os detalhes das publicações, como nome, veículo associado, categoria, ano, Qualis e pontuação. Apesar do método `FrameWeb` não

ter suporte para representação de modais, foi criado um estereótipo «dialog» para tal representação como sugestão para melhoria do método.

Na Figura 16 é apresentado o fluxo para o caso de uso **Extrair Informação de Qualificação**, onde as qualificações de veículos são extraídas e armazenadas, conforme Qualis associada e o ano o qual o arquivo a ser lido se refere.

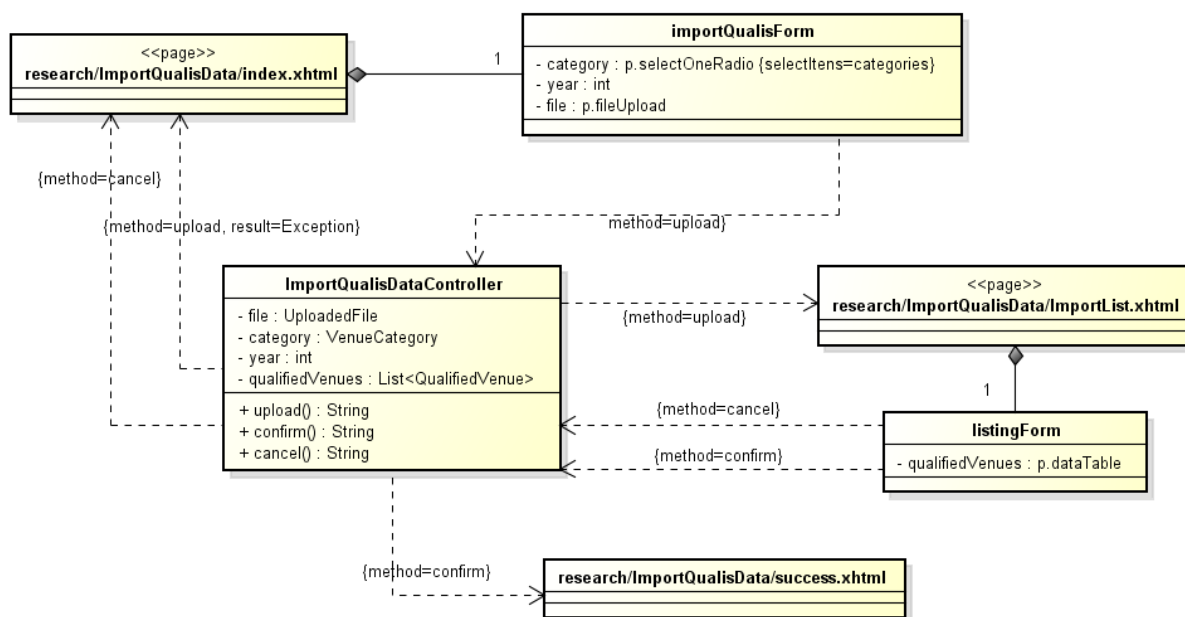


Figura 16 – Modelo de Navegação para fluxo de Extrair Informação de Qualificação.

Primeiramente, preenche-se o formulário `importQualisForm` contido na página inicial `index.xhtml`, informando o ano de referência e marcando a categoria (conferência ou periódico). Quando `upload()` é executado, o usuário é direcionado então para a página `importList.xhtml`, a qual possui um formulário, `listingForm`, o qual mostra em formato de tabela os dados que foram lidos do arquivo CSV. Ao confirmar a ação, o sistema redireciona para a página de sucesso, por meio do método `confirm`. Se o fluxo for executado de forma satisfatória, o usuário é redirecionado para a página `success.xhtml`, uma página usada apenas para mostrar ao usuário que o processo foi realizado. Caso contrário, o sistema retorna à página inicial, gerando uma exceção (`Exception`). O método `cancel` redireciona o usuário para a página inicial, para um novo cálculo, se assim desejar.

4.2.4 Modelo de Aplicação

O **Modelo de Aplicação** é um diagrama UML que representa as classes de serviço, classes estas responsáveis pela codificação dos casos de uso, e suas dependências. Este diagrama é usado para orientar na implementação das classes do pacote Aplicação e a configuração das dependências entre os pacotes Controle e Aplicação e pacotes Aplicação e Persistência (SOUZA, 2007).

Todas as classes de aplicação que são de casos de uso cadastrais estendem de *CrudService*, a qual é representada na Figura 17 de forma simplificada. Por convenção do *FrameWeb*, os métodos da classe não são repetidos na interface, pois considera-se que a interface declara todos os métodos públicos da classe que a implementa.

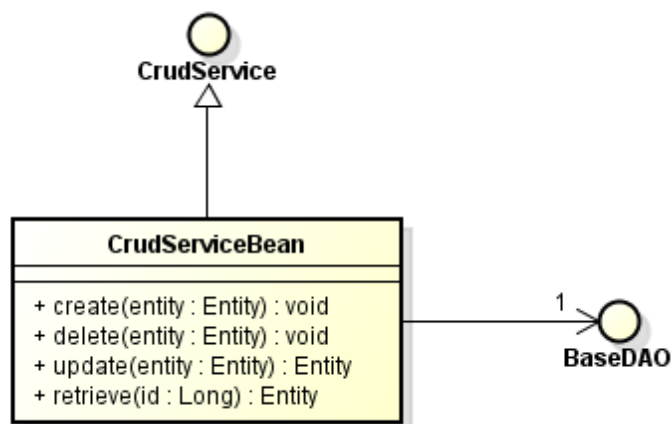


Figura 17 – Classe *CrudServiceBean* e métodos CRUD.

As figuras 18 e 19 representam o **Modelo de Aplicação** do módulo C2D. Nelas estão representadas as classes de aplicação usadas no C2D e as dependências que existem entre a camada de controle, camada de aplicação e a camada de persistência.

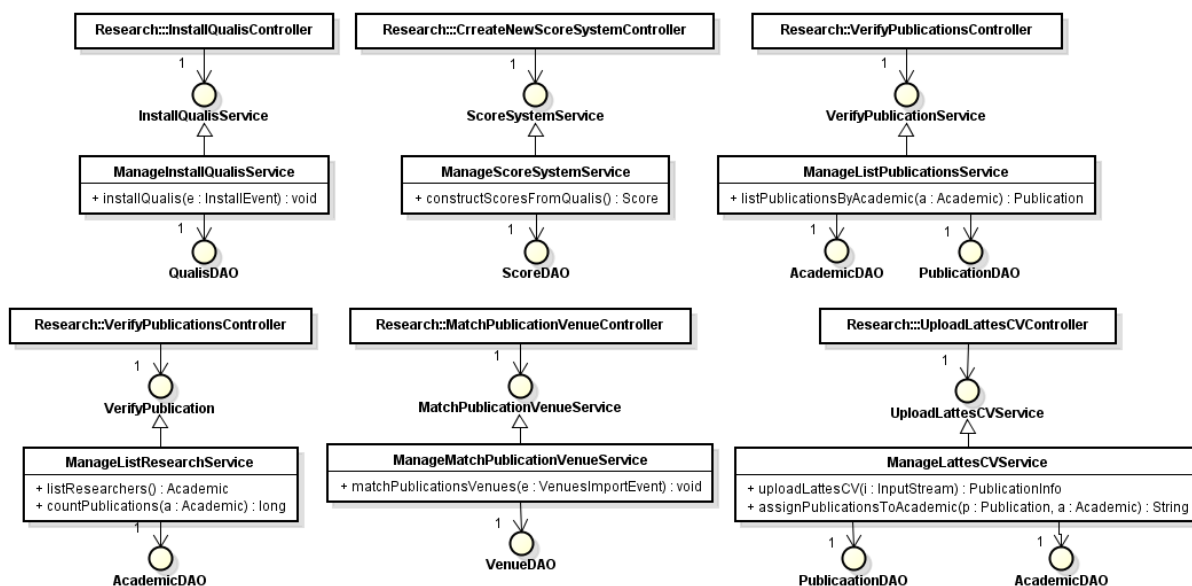


Figura 18 – Modelo de Aplicação do sistema C2D, primeira parte.

Por exemplo, na Figura 18 é apresentada a classe *ManageLattesCVService*. Esta classe se comunica com a classe de controle *UploadLattesCVController* e com os DAO's de *Publication* e *Academic*, os quais são responsáveis por persistir os dados de suas respectivas classes. O mesmo acontece com *ImportQualisDataService*, a qual possui associação com a classe de controle *ImportQualisDataController* e com *VenueDAO*.

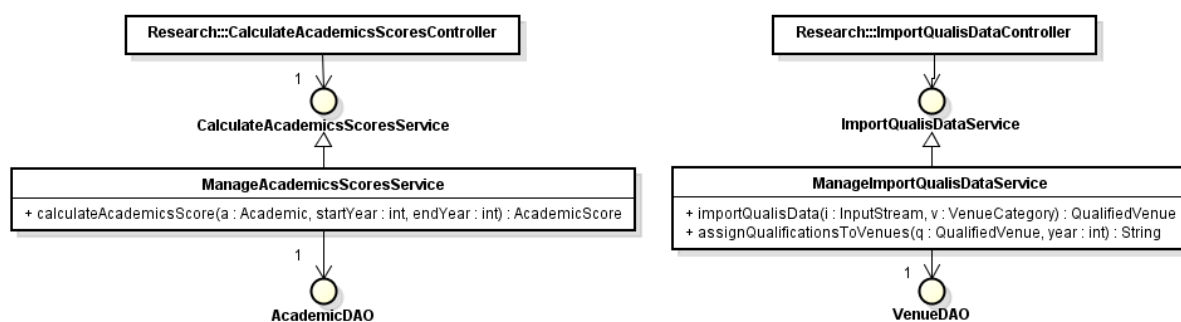


Figura 19 – Modelo de Aplicação do sistema C2D, segunda parte.

5 Apresentação

Nesta seção são apresentadas diversas capturas de tela do sistema, a fim de demonstrar o resultado obtido.

5.1 Login

A Figura 20 apresenta a tela inicial do sistema, contendo somente a opção de login. Ao clicar nesta opção, é mostrada a tela de login do sistema, como mostra a Figura 21 com os campos de login e senha. As telas de Login e ManageAcademics já estavam implementadas quando este projeto teve início, porém são mostradas para que a sequência de telas não fique incompleta e sem sentido. Como dito, para efetuar login é necessário email e senha. É feita uma verificação do email para confirmar se é um email válido que está sendo inserido. O campo da senha aceita qualquer caractere alfanumérico.



Figura 20 – Tela inicial do sistema C2D.

Do lado esquerdo da tela é possível ver um menu, inicialmente contendo apenas a funcionalidade de login. Efetuado o acesso, o usuário poderá ver as funcionalidades previamente existentes no sistema Marvin (Figura 22) e aquelas implementadas para o módulo C2D (Figura 23).

5.2 Upload LattesCV

Na Figura 24 é mostrada a tela de Upload Lattes CV. Essa função permite que a secretária extraia os dados contidos no currículo Lattes de um acadêmico de forma

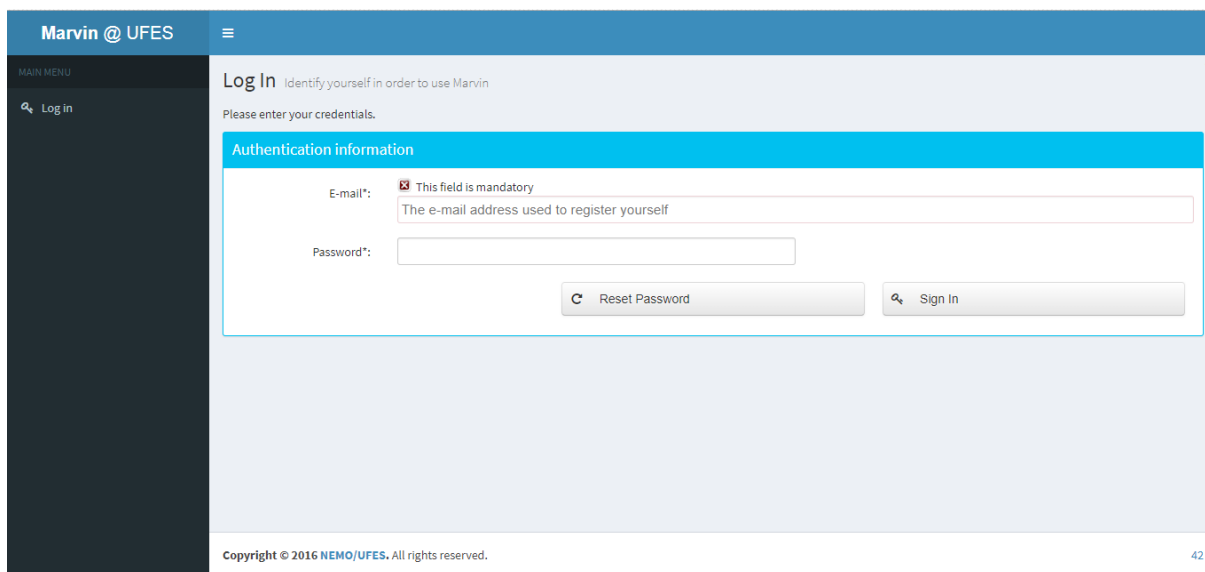


Figura 21 – Tela de login do sistema C2D.



Figura 22 – Funcionalidades já existentes no Marvin.

automática. Para isso, basta clicar no botão **Choose** deve-se escolher o currículo desejado, anteriormente baixado, no formato XML e clicar em **Upload**. Caso não haja nenhum acadêmico com o identificador do currículo selecionado, o sistema retorna uma mensagem de aviso, como mostra a Figura 25. O processo de extração de dados do currículo Lattes ocorre de forma assíncrona, ou seja, a secretária pode realizar outra atividade enquanto o primeiro está sendo feito, desde que não seja algo que necessite destes dados no momento da extração.

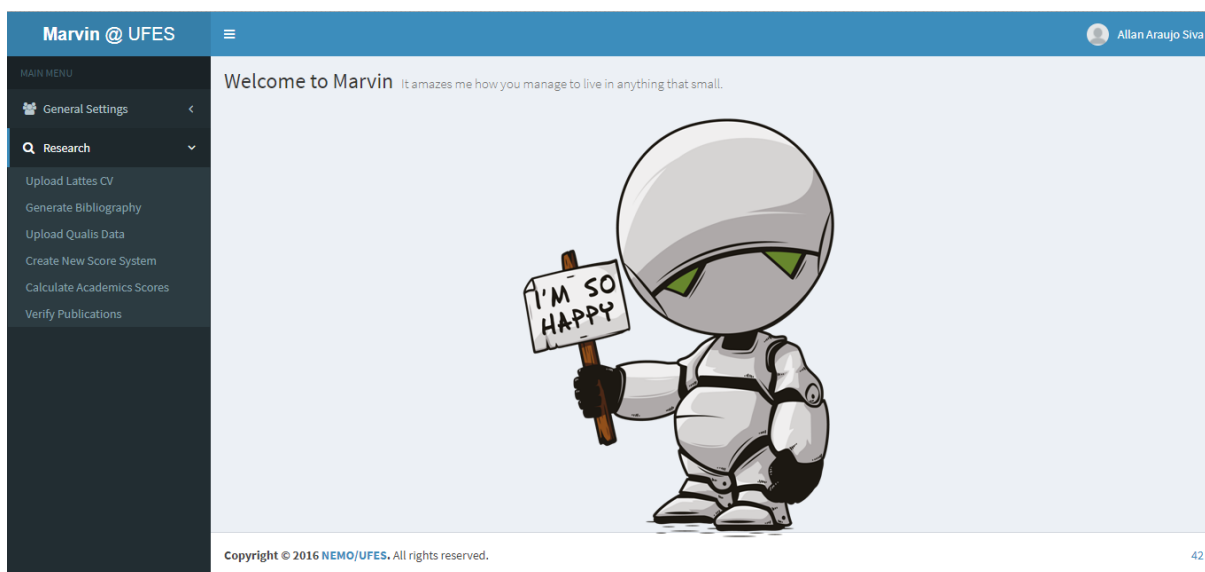


Figura 23 – Funcionalidades implementadas para o C2D.

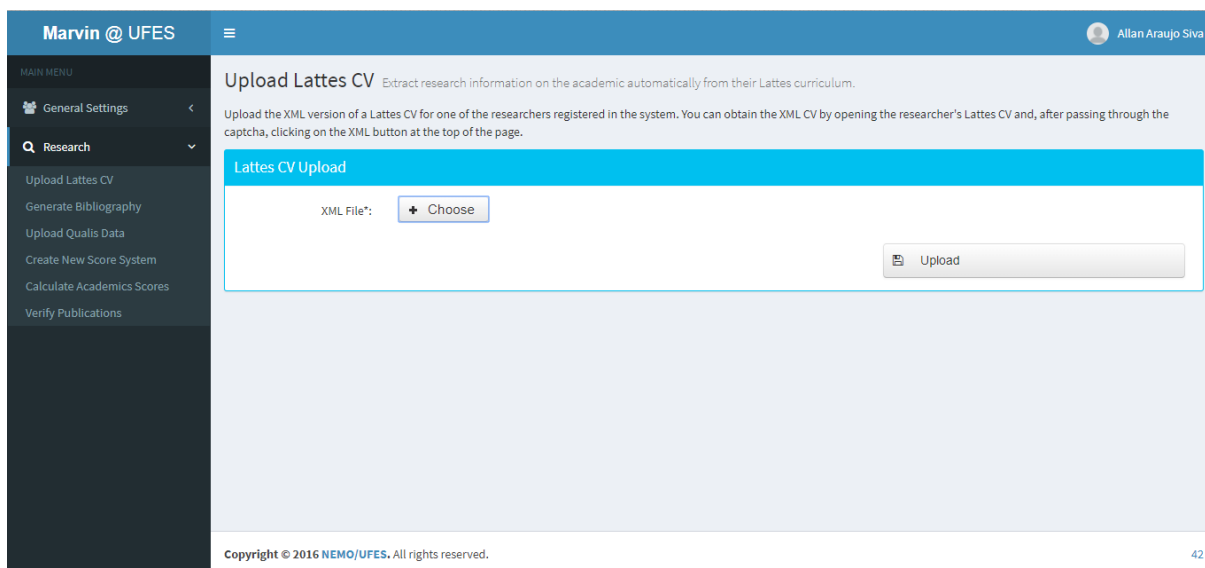


Figura 24 – Upload Lattes CV - C2D.

5.3 UploadQualis Data

Na Figura 26 é mostrada a tela de Upload Qualis Data. É feita aqui a extração dos dados de Qualis dos veículos cadastrados no arquivo de classificação. Este arquivo é diferenciado de acordo com o tipo de veículo: conferência ou periódico. Caso se queira extrair dados de arquivo de classificação de conferência, marca-se o *radio button* referente à **Conference**. No caso de periódico, marca-se **Jounal**. No campo **year**, informa-se o ano de referência. Para escolher o arquivo, basta clicar no botão **Choose**. O arquivo deve estar no formato CSV, sendo que:

- Arquivo para conferências possui os campos: Sigla, Nome, Índice-H e Estrato (Qualis).

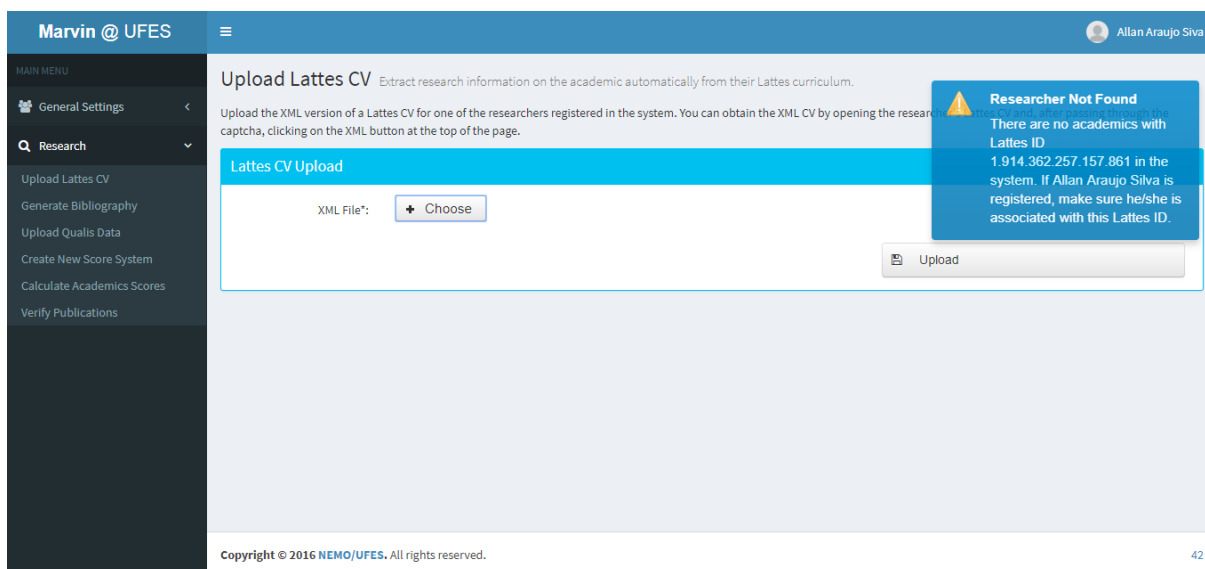


Figura 25 – Upload Lattes CV - C2D Aviso.

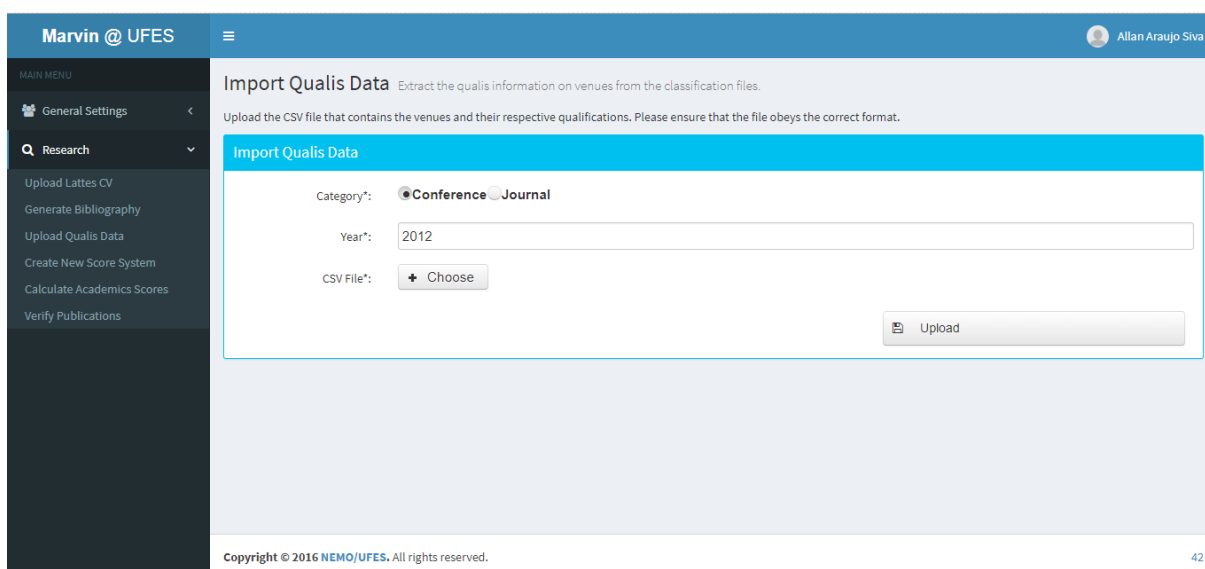


Figura 26 – Upload Qualis Data - C2D.

- Arquivo para periódicos possui os campos: ISSN, Título e Estrato (Qualis);

Caso o arquivo não esteja de acordo com o formato especificado, o sistema retorna uma mensagem de erro, como mostrado na Figura 27.

As Figuras 28 e 29 mostram um exemplo de extração de dados para arquivo de configuração de periódicos referente ao ano de 2016. O sistema permite que a secretária visualize os dados antes de realizar a extração. Como os arquivos costumam ser grandes, foi utilizado o esquema de paginação, para melhor organização. Feito isso, basta clicar em OK para finalizar o processo ou Cancel, para abortar o processo.

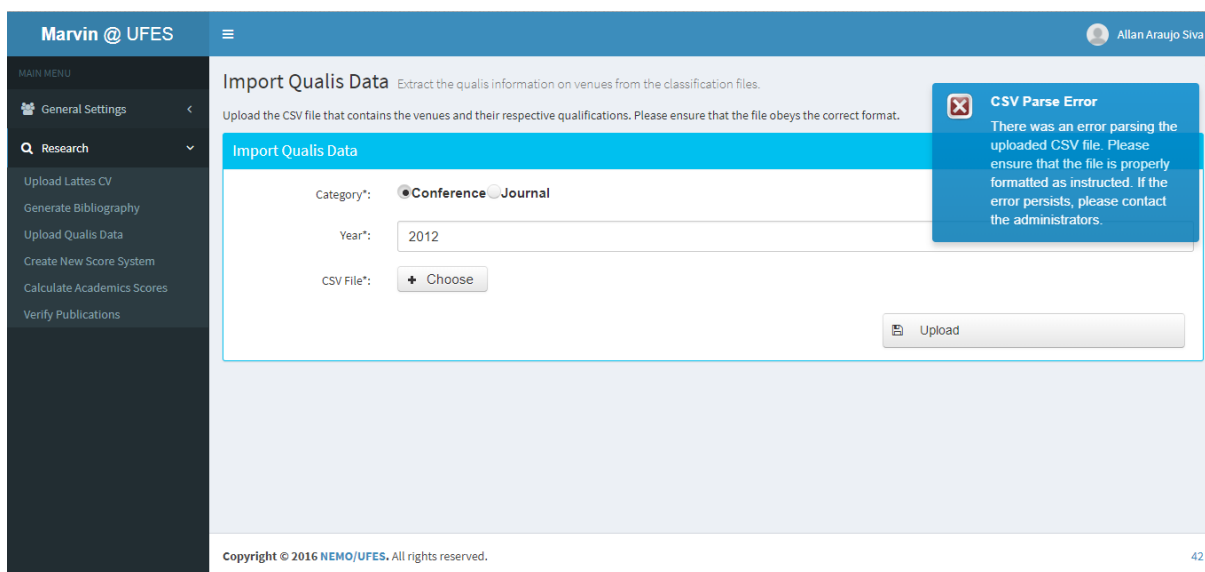


Figura 27 – Upload Qualis Data - C2D - erro de parsing.

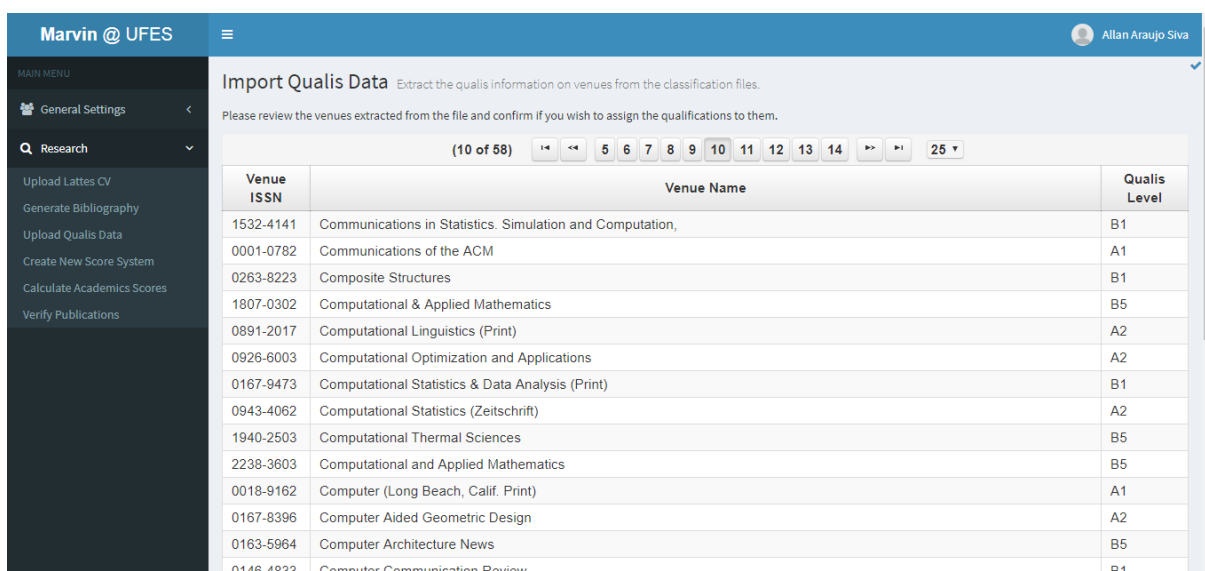


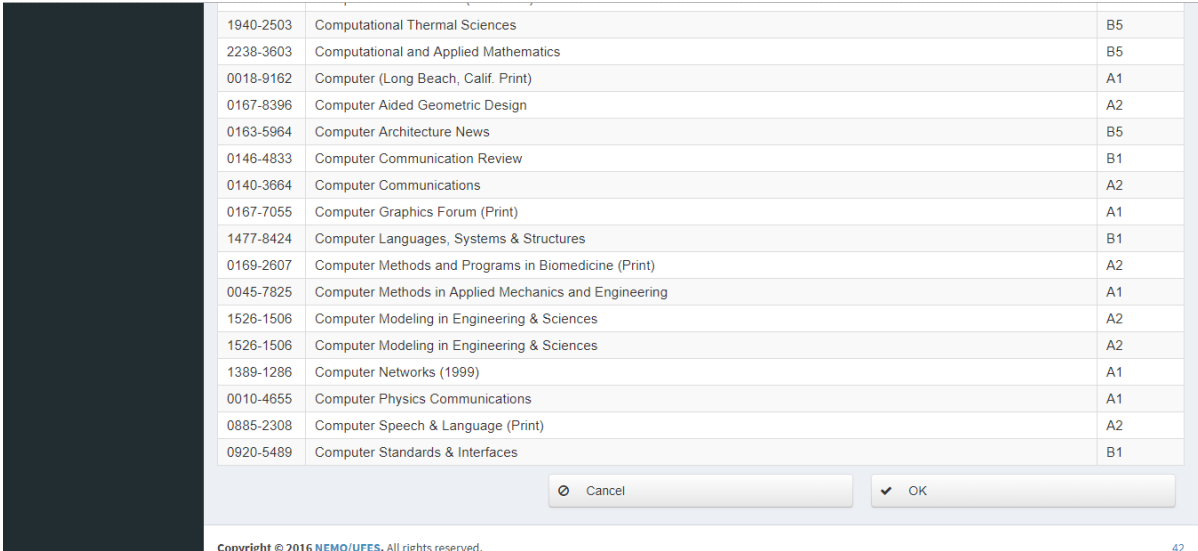
Figura 28 – Upload Qualis Data - C2D - Journal.

5.4 Create New Score System

A Figura 30 apresenta a tela onde cria-se o sistema de pontuação para conferência e periódico. Para cada classe de Qualis, a secretária insere a pontuação desejada. As Qualis possuem seus campos restritos para receber um número, impedindo que sejam inseridos caracteres inválidos.

5.5 Calculate Academics Score

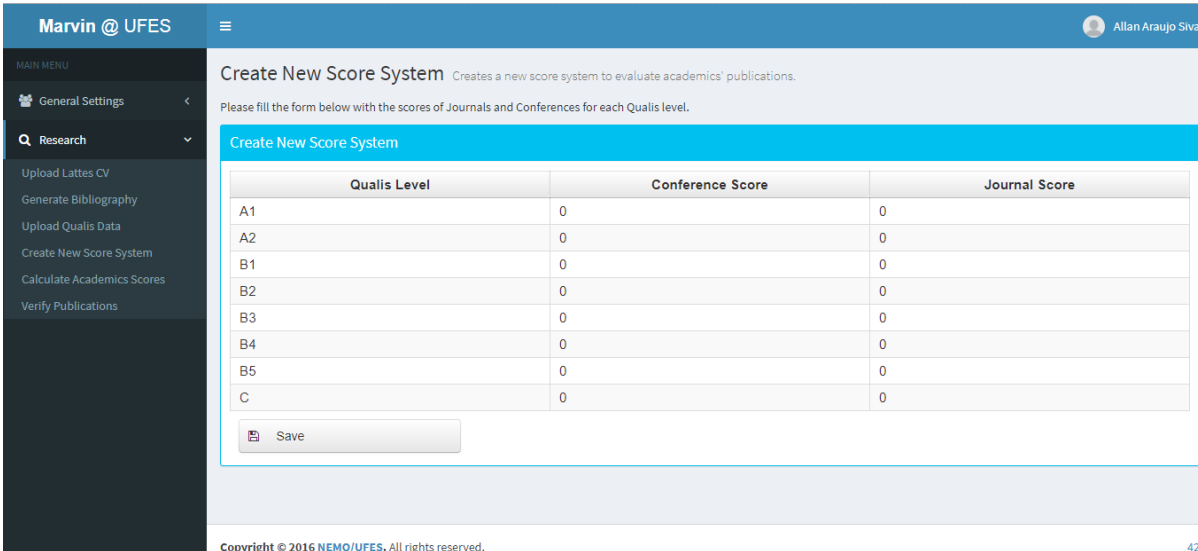
A Figura 31 mostra a tela de cálculo de pontuação dos docentes cadastrados. Foram criados alguns cadastros para realização de testes no sistema, como apresentado na figura.



1940-2503	Computational Thermal Sciences	B5
2238-3603	Computational and Applied Mathematics	B5
0018-9162	Computer (Long Beach, Calif. Print)	A1
0167-8396	Computer Aided Geometric Design	A2
0163-5964	Computer Architecture News	B5
0146-4833	Computer Communication Review	B1
0140-3664	Computer Communications	A2
0167-7055	Computer Graphics Forum (Print)	A1
1477-8424	Computer Languages, Systems & Structures	B1
0169-2607	Computer Methods and Programs in Biomedicine (Print)	A2
0045-7825	Computer Methods in Applied Mechanics and Engineering	A1
1526-1506	Computer Modeling in Engineering & Sciences	A2
1526-1506	Computer Modeling in Engineering & Sciences	A2
1389-1286	Computer Networks (1999)	A1
0010-4655	Computer Physics Communications	A1
0885-2308	Computer Speech & Language (Print)	A2
0920-5489	Computer Standards & Interfaces	B1

Copyright © 2016 NEMO/UFES. All rights reserved. 42

Figura 29 – Upload Qualis Data - C2D - Confirmação.



Marvin @ UFES Allan Araujo Siva

MAIN MENU

- General Settings
- Research
 - Upload Lattes CV
 - Generate Bibliography
 - Upload Qualis Data
 - Create New Score System
 - Calculate Academics Scores
 - Verify Publications

Create New Score System Creates a new score system to evaluate academics' publications.

Please fill the form below with the scores of Journals and Conferences for each Qualis level.

Qualis Level	Conference Score	Journal Score
A1	0	0
A2	0	0
B1	0	0
B2	0	0
B3	0	0
B4	0	0
B5	0	0
C	0	0

Save

Copyright © 2016 NEMO/UFES. All rights reserved. 42

Figura 30 – Create New Score System - C2D.

Para realizar o cálculo, basta indicar o período desejado: ano inicial no campo **Start Year** e ano final no campo **End Year**. O sistema permite a escolha de um ou mais acadêmicos para calcular a pontuação. O cálculo é feito de acordo com as pontuações de Qualis cadastradas no sistema de pontuação e com as classificações das publicações do docente do qual deseja-se saber a pontuação. Basta clicar em **Calculate** para prosseguir ou clicar em **Star Over** para retroceder.

A Listagem 5.1, mostra o código fonte do método `calculateAcademicsScore`. Neste processo, é efetuado o *matching* entre veículos Lattes e veículos Qualis para que seja feito o cálculo da pontuação do docente selecionado.

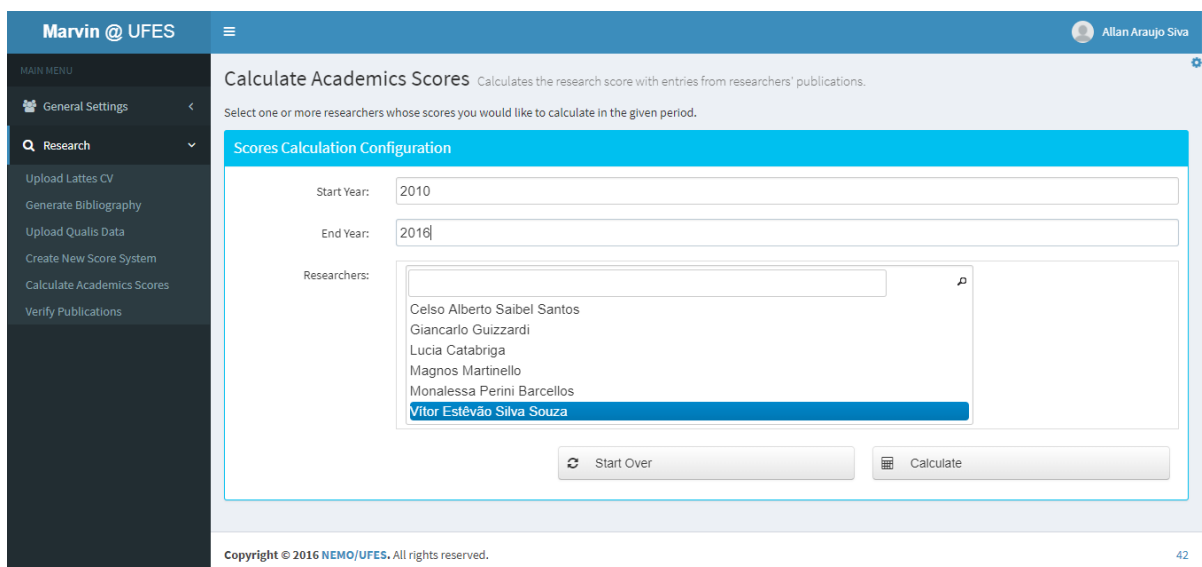


Figura 31 – Calculate Academics Score - C2D.

Listagem 5.1 – Código que efetua o cálculo da pontuação de um acadêmico.

```

1 public List<AcademicScore> calculateAcademicsScore(List<Academic> academics, int
2   startYear, int endYear) throws ScoreSystemNotRegisteredException {
3   try {
4     //Retrieve the current scores
5     ScoreSystem currentScoreSystem = scoreSystemDAO.
6       retrieveCurrentScoreSystem();
7     List<Score> currentScoresList = scoreDAO.retrieveByScoreSystem(
8       currentScoreSystem);
9     Map<Qualis, Score> scoreQualisMap = new HashMap<Qualis, Score>();
10    for(Score s : currentScoresList) {
11      scoreQualisMap.put(s.getQualis(), s);
12    }
13
14    List<AcademicScore> academicScoreList = new ArrayList<AcademicScore>();
15    for(Academic a : academics) {
16      AcademicScore as = new AcademicScore();
17      as.setAcademic(a);
18
19      List<Publication> publicationsList = publicationDAO.
20        retrieveByAcademicAndYearRange(a, startYear, endYear);
21      List<PublicationScore> publicationScoreList = new ArrayList<
22        PublicationScore>();
23
24      //Stores the academic score in each category
25      int scoreConferenceAcademic = 0;
26      int scoreJournalAcademic = 0;
27
28      for(Publication p : publicationsList) {
29        Venue pubVenue = p.getVenue();
30
31        //No way to calculating the score of publications without
32        //associated venues
33        if (pubVenue == null) continue;
34        try {

```



```

29         Qualification quaPubVenue = qualificationDAO .
30             retrieveClosestByVenueAndYear(pubVenue, p.getYear());
31         PublicationScore publicationScore = new PublicationScore();
32         publicationScore.setPublication(p);
33
34         //Stores the current publication score
35         int currentScore = 0;
36         boolean venueIsConference = pubVenue.getCategory().equals(
37             VenueCategory.CONFERENCE);
38
39         Qualis qualis = quaPubVenue.getQualis();
40         Score score = scoreQualisMap.get(qualis);
41         publicationScore.setQualis(qualis);
42
43         if (venueIsConference) {
44             currentScore += score.getScoreConference();
45             scoreConferenceAcademic += currentScore;
46         }
47         else {
48             currentScore += getScoreJournal();
49             scoreJournalAcademic += currentScore;
50         }
51         publicationScore.setScore(currentScore);
52         publicationsList.add(publicationScore);
53     }
54     catch(PersistentObjectNotFoundException e) {
55         // If there is no qualification that applies to the current
56         publication ,
57         // skip this publication and go to the next one;
58         logger.log(Level.WARNING,
59             "No qualification from an year that is less than or
60             equal to the publication's year was found.");
61     }
62     catch (MultiplePersistentObjectsFoundException e) {
63         // This is a bug. Log and throw a runtime exception.
64         logger.log(Level.SEVERE, "Multiple qualifications found that
65             have the same year and belong to the same venue.", e);
66         throw new EJBException(e);
67     }
68 }
69 as.setPublicationsScores(publicationScoreList);
70 as.setScoreConference(scoreConferenceAcademic);
71 as.setScoreJournal(scoreJournalAcademic);
72 as.setScoreTotal(scoreConferenceAcademic + scoreJournalAcademic);
73 academicScoreList.add(as);
74 }
75 return academicScoreList;
76
77 } catch (PersistentObjectNotFoundException e) {
78     // If there is no Score System that is currently active, throw an
79     // exception from the domain.
80     logger.log(Level.WARNING,
81         "No currently active score system was found.");
82     throw new ScoreSystemNotRegisteredException();

```

```
81     } catch (MultiplePersistentObjectsFoundException e) {
82         // This is a bug. Log and throw a runtime exception.
83         logger.log(Level.SEVERE, "Multiple score systems found that are currently
            active.", e);
84         throw new EJBException(e);
85     }
86 }
```

Primeiramente, recupera-se as informações do `ScoreSystem` atual, as quais são armazenadas em um dicionário (`HashMap`) que associa cada `Qualis` à sua pontuação. Então, para cada acadêmico, é associado um `AcademicScore`, objeto utilizado para guardar as pontuações de periódico, conferência e total de determinado acadêmico. Em seguida, as publicações do acadêmico no período em questão são recuperadas e uma lista de `PublicationScore` é criada para guardar a pontuação de cada publicação.

Então, para cada publicação existente na lista de publicações do acadêmico, o veículo é verificado. Caso exista um veículo associado à publicação, o sistema recupera uma qualificação de acordo com a publicação e com o ano de referência, guardando no atributo `quaPubVenue`. Guarda-se o `Qualis` do veículo no `HashMap`, obtendo a pontuação da publicação. É verificada então a categoria do veículo (conferência ou periódico) incrementando a pontuação da respectiva categoria. O total é simplesmente a soma das duas pontuações. Após calcular a pontuação da publicação, esta é adicionada à lista de publicações do acadêmico.

Caso não exista um veículo nesta publicação, não há como o sistema fazer a associação de `Qualis` da publicação e o cálculo da pontuação. Neste caso, a publicação fica sem pontuação. Após serem analisadas todas as publicações, as informações são armazenadas no objeto `AcademicScore` e o algoritmo continua com o próximo acadêmico da lista. Ao concluir todos os acadêmicos, o algoritmo retorna a lista de `AcademicScore` para que os resultados sejam apresentados ao usuário.

A Figura 32 mostra um exemplo de cálculo das pontuações de conferência, periódico e o total de determinado acadêmico. Ao clicar na linha, é mostrada uma nova janela (estilo *modal*, vide Figura 33) onde os detalhes de cada publicação são listados. Tais detalhes são: Nome da publicação, veículo associado, categoria, ano, `Qualis` e pontuação. Assim, a secretária consegue visualizar as publicações e as pontuações associadas, bem como as publicações que não tiveram pontuação (representadas com a coluna `Score = Not Found`).

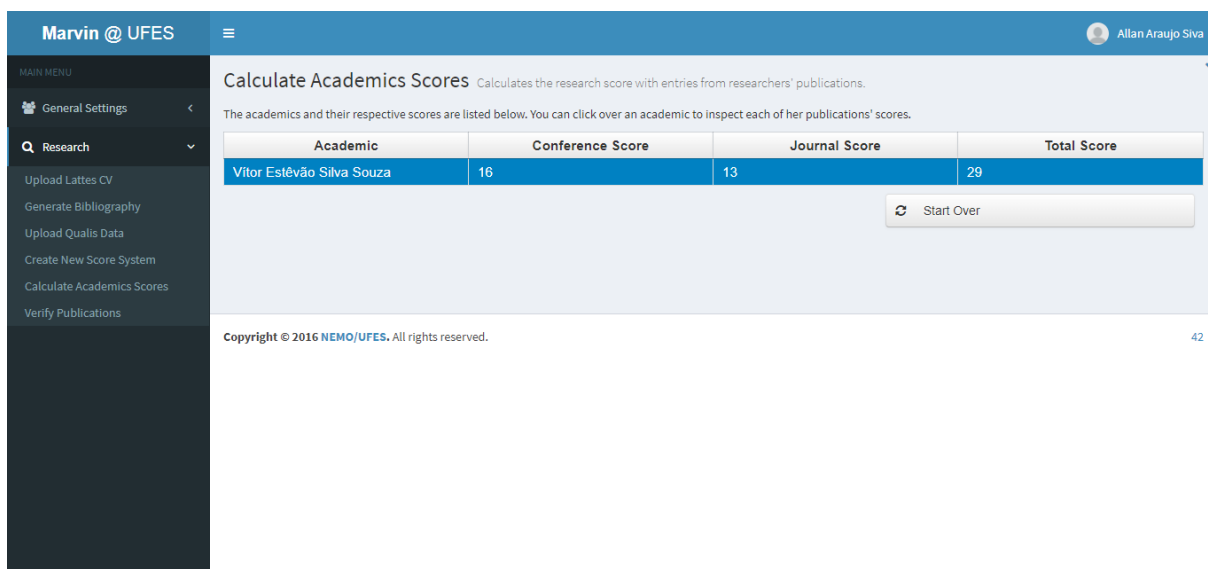


Figura 32 – Calculate Academics Score - C2D - Resultado.

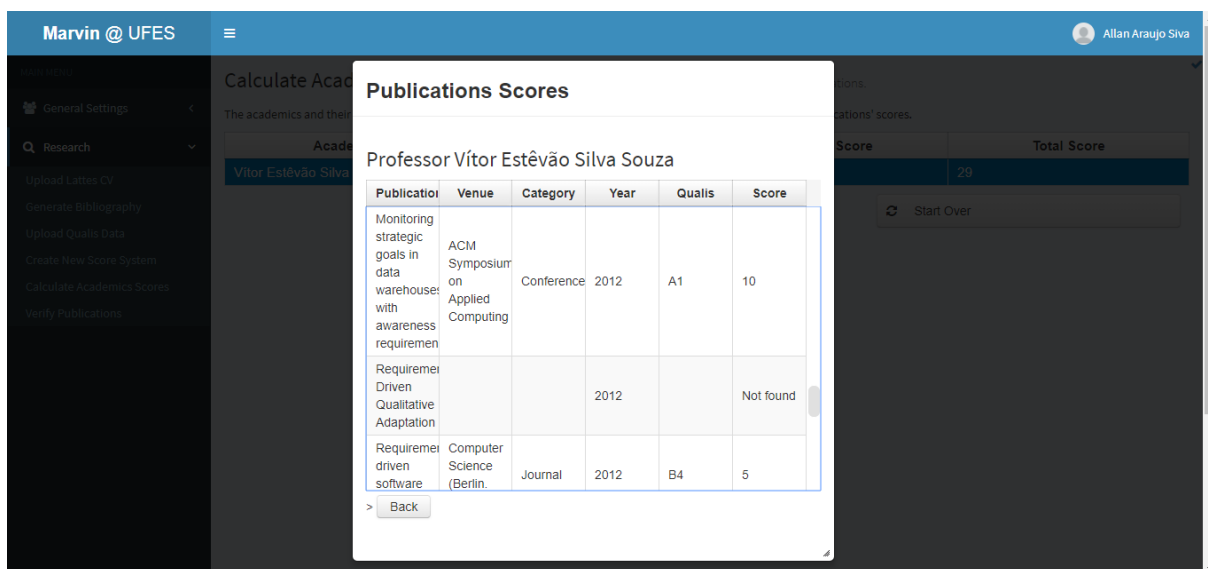


Figura 33 – Calculate Academics Score - C2D - Detalhes.

6 Considerações Finais

Neste capítulo tem-se as conclusões do trabalho realizado, bem como as contribuições geradas a partir dele. Além disso, são apresentadas suas limitações e perspectivas de possíveis trabalhos futuros.

6.1 Conclusão

Devido à importância do registro de dados de docentes no PPGI/UFES, seu devido gerenciamento e visto que não havia esse tipo de sistema no ambiente do DI/UFES, surgiu a necessidade da criação de um sistema/módulo que permitisse a execução destas atividades, além de integrar as ferramentas desenvolvidas ao sistema já existente de maneira que sejam realmente usadas.

Seguindo o roteiro proposto no Capítulo 1, os objetivos que foram listados foram alcançados. Objetivos foram identificados para melhor modelagem do sistema e compreensão do problema, utilizando a linguagem iStar. Requisitos foram levantados e analisados para a elaboração dos documentos de requisitos e de projetos, segundo critérios e padrões utilizados na Engenharia de Software. Feito o documento de requisitos, o documento de projeto foi então elaborado, contendo as informações sobre a arquitetura do sistema e os modelos propostos utilizando a abordagem FrameWeb. Por fim, um protótipo do sistema foi implementado.

Em paralelo, foi desenvolvido um trabalho sobre sistemas Web que trabalham com dados interligados (*Linked Data*) e riscos envolvidos nestes processo, onde o aluno de mestrado do PPGI/UFES Marcio Louzada de Freitas propôs os modelos de riscos e modelos de objetivos para o sistema C2D, a fim de mapear os riscos no compartilhamento e gerenciamento de dados interligados. Neste trabalho, participei como colaborador, auxiliando na modelagem de riscos e na análise de objetivos. Estes modelos são usados para melhor representação dos processos existentes no sistema e como cada um influencia no conjunto total da aplicação. Com os resultados obtidos no trabalho, foi redigido um artigo (FREITAS et al., 2018, em avaliação), o qual foi submetido ao CIBSE,¹ *Requirements Engineering track*, conferência cujo objetivo é promover pesquisas científicas de alta qualidade nos países ibero-americanos, apoiando os pesquisadores desta comunidade na publicação e discussão dos trabalhos.

Uma das maiores dificuldades enfrentadas foi adaptar o conhecimento existente em desenvolvimento web e implementar um sistema utilizando Java EE e suas tecnologias

¹ <<http://cibseconference.org/>>

(JSF, CDI, JPA). Para isso, foi realizada a revisão bibliográfica, com leitura de artigos e tutoriais de desenvolvimento utilizando Java indicados ao longo do projeto e orientação dada pelo professor. Além disso, foi preciso estudar o método `FrameWeb`, visto que não é um conceito fácil de ser consolidado em tão pouco tempo, principalmente no que diz respeito à construção dos modelos

Fato é que o conhecimento adquirido ao longo da graduação proporcionou uma base para que este projeto fosse desenvolvido. Já citadas anteriormente, as disciplinas de Engenharia de Software e Requisitos, Programação, Lógica Computacional, Banco de Dados, Desenvolvimento Web e outras as quais não foram citadas foram de suma importância para o aprendizado e formação acadêmica.

6.2 Limitações e Trabalhos Futuros

Em qualquer software, melhorias como criação de novas funcionalidades e ajustes sempre ocorrem. Com isso, é necessário que haja constante manutenção do sistema, sendo capaz de identificar problemas, verificar tecnologias que se tornaram depreciadas e não são mais utilizadas, para que o sistema permaneça o mais atualizado possível. A partir dos resultados obtidos, podem ser citadas algumas limitações que o sistema apresenta. Tais limitações possibilitam que novos trabalhos e projetos possam ser realizados. A seguir, são listadas algumas limitações:

- Aprimorar a forma de *matching* entre veículo declarado no Lattes e veículo declarado no Qualis, visto que as pessoas podem escrever o nome do veículo diferente da forma como o Qualis apresenta;
- Aprimorar a seleção de Qualis (permitir avaliar segundo Qualis posterior ou anterior);
- Aprimorar a forma como é gerado o arquivo de pontuação, oferecendo opção de seleção de campos no arquivo e habilitando o download do arquivo em diferentes formatos;
- Aprimorar a interface com o usuário, em especial as funcionalidades que possuem processamento assíncrono, pois atualmente não notificam o usuário quando são concluídas.

Referências

- ASSOCIATION, I. S. et al. Standard glossary of software engineering terminology. *IEEE Std*, p. 610–12, 1990. Citado na página 14.
- AURUM, A. et al. *Managing software engineering knowledge*. [S.l.]: Springer Science & Business Media, 2013. Citado 2 vezes nas páginas 14 e 15.
- DALPIAZ, F.; FRANCH, X.; HORKOFF, J. istar 2.0 language guide. *CoRR*, abs/1605.07767, 2016. Disponível em: <<http://arxiv.org/abs/1605.07767>>. Citado na página 15.
- FALBO, R. d. A. *Engenharia de Software*. [s.n.], 2014. 144 p. Disponível em: <https://inf.ufes.br/~falbo/files/ES/Notas_Aula_Engenharia_Software.pdf>. Citado na página 13.
- FALBO, R. d. A. *Projeto de Sistemas*. [s.n.], 2016. 138 p. Disponível em: <https://inf.ufes.br/~falbo/files/PSS/Notas_Aula_Projeto_Sistemas_2016.pdf>. Citado na página 17.
- FALBO, R. d. A. *Engenharia de Requisitos*. [s.n.], 2017. 178 p. Disponível em: <https://inf.ufes.br/~falbo/files/ER/Notas_Aula_Engenharia_Requisitos.pdf>. Citado 2 vezes nas páginas 14 e 15.
- FARIA, T. *Java EE 7 com JSF, PrimeFaces e CDI*. [S.l.: s.n.], 2013. Citado na página 19.
- FOWLER, M. *Patterns of enterprise application architecture*. [S.l.]: Addison-Wesley Longman Publishing Co., Inc., 2002. Citado 2 vezes nas páginas 18 e 22.
- FREITAS, M. L. de et al. Goal and Risk Analysis in the Development of Information Systems for the Web of Data. In: *submetido à 21st Ibero-American Conference on Software Engineering, Requirements Engineering track*. [S.l.: s.n.], 2018, em avaliação. Citado na página 51.
- GINIGE, A.; MURUGESAN, S. Web engineering: An introduction. *IEEE multimedia*, IEEE, v. 8, n. 1, p. 14–18, 2001. Citado na página 19.
- KOTONYA, G.; SOMMERVILLE, I. *Requirements engineering: processes and techniques*. [S.l.]: Wiley Publishing, 1998. Citado 2 vezes nas páginas 6 e 14.
- LIMA, L. V. F. *SAP - Sistema de Apoio ao Professor*. [S.l.], 2015. Citado 2 vezes nas páginas 6 e 31.
- MANZOLI, B. *SAE - Sistema de Acompanhamento de Egressos*. [S.l.], 2016. Citado na página 19.
- MARTINS, B. F. S. *Uma abordagem dirigida a modelos para o projeto de Sistemas de Informação Web com base no método FrameWeb*. Dissertação (Mestrado) — Universidade Federal do Espírito Santo, 2016. Citado na página 22.

MYLOPOULOS, J.; CHUNG, L.; YU, E. From object-oriented to goal-oriented requirements analysis. *Commun. ACM*, ACM, New York, NY, USA, v. 42, n. 1, p. 31–37, jan. 1999. ISSN 0001-0782. Disponível em: <<http://doi.acm.org/10.1145/291469.293165>>. Citado na página 15.

PRESSMAN, R. S. *Engenharia de software*. [S.l.]: Makron books Sao Paulo, 2011. v. 7. Citado 2 vezes nas páginas 13 e 17.

SALVATORE, T. R. *AlocaWeb e BibLattes - Módulos do sistema Marvin*. Monografia (Projeto de Graduação) — Universidade Federal do Espírito Santo, 2016. Citado 4 vezes nas páginas 6, 22, 36 e 37.

SOUZA, V. E. S. *FrameWeb – A Framework-based Design Method for Web Engineering*. Dissertação (Mestrado) — Universidade Federal do Espírito Santo, 2007. Citado 4 vezes nas páginas 11, 22, 35 e 39.

Apêndices



Documento de Especificação de Requisitos

C2D - Sistema de Credenciamento e Classificação de Docentes

Registro de Alterações:

Versão	Responsável	Data	Alterações
1.0.0	Allan Araujo Silva	05/07/2017	Versão inicial da documentação

Vitória, ES

2017

1 Introdução

Este documento apresenta os requisitos de usuário do Sistema de Credenciamento e Classificação de Docentes do PPGI (C2D) e está organizado da seguinte forma: a Seção 2 contém uma descrição do propósito do sistema; a Seção 3 apresenta uma descrição do minimundo apresentando o problema; e a Seção 4 apresenta as listas de requisitos de usuário levantados junto ao cliente.

Na Seção 2 deste documento descreve-se superficialmente o sistema, apresentando de forma geral suas principais características. Já na Seção 3 estão apresentadas as funcionalidades do C2D, bem como suas dependências. A Seção 4 lista os requisitos de usuário do sistema (funcionais, não funcionais e regras de negócio). A Seção 5 apresenta o modelo de casos de uso, incluindo descrições de atores, os diagramas de casos de uso e suas respectivas descrições. Na Seção 6 estão apresentados os modelos conceituais estruturais do sistema na forma de diagramas de classes. Por fim, a Seção 7 apresenta o dicionário do projeto, contendo as definições das classes identificadas.

2 Descrição do Propósito do Sistema

O sistema tem como propósito realizar as principais atividades relacionadas ao credenciamento e gerenciamento de docentes do Programa de Pós-Graduação em Informática (PPGI) da Universidade Federal do Espírito Santo (Ufes), controlando pontuação de publicações necessárias para entrar no programa como docente. Para que essas atividades sejam realizadas, é necessário buscar informações no Currículo Lattes na plataforma Web lattes.cnpq.br e também em bancos de dados Web em formato de dados interligados.

3 Descrição do Minimundo

Programas de Pós-Graduação são avaliados pela Capes (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) de acordo com sua produção técnica, ou seja, publicação de artigos científicos em veículos (conferências e periódicos) qualificados. Por esta razão, os programas de pós-graduação (PPGs) elaboram critérios para que docentes sejam credenciados no programa, se mantenham dentro do programa, alcancem diferentes níveis dentro do programa, etc. Todos estes critérios giram em torno das publicações dos professores e da qualificação que a Capes faz dos veículos onde estes artigos são publicados, chamada de Qualis.

O C2D deve considerar, inicialmente, os critérios estabelecidos pelo PPGI/Ufes, mas sendo também genérico o suficiente para permitir atualização destes critérios (caso mudem no futuro), podendo, assim, também ser aproveitado por outros PPGs. Para efeito de Minimundo, no entanto, consideram-se os critérios de publicação atuais do PPGI.

O PPGI define atualmente 4 critérios: (a) ingresso no programa; (b) credenciamento e recredenciamento no programa; (c) categorização como colaborador ou permanente; e (d) habilitação para orientação no doutorado. Tais critérios envolvem vários aspectos, como: ter plano de trabalho aprovado, ter lecionado disciplinas, ter concluído orientações, ter vínculo funcional-administrativo e, por fim, atender a requisitos mínimos de publicação.

A princípio, o **C2D deve avaliar apenas os requisitos de publicação**. Neste contexto, os critérios (a) e (b) utilizam um requisito de publicação, enquanto o critério (d) utiliza um outro requisito de publicação. O requisito para ingresso/permanência no programa é somar 10 pontos de publicação no último triênio, enquanto o requisito para orientar no doutorado é somar 25 pontos, sendo que 10 destes pontos devem resultar de publicações em periódicos.

3.1 Cálculo de Pontuação

Tal pontuação é calculada de acordo com a classificação do veículo de publicação (conferência ou periódico) no Qualis da Capes. Atualmente, a pontuação é dada seguindo as regras abaixo:

- Periódicos internacionais com fator de impacto equivalente* a A1, A2 ou B1: 20 pontos;
- Periódicos internacionais com fator de impacto equivalente a B2 ou B3: 10 pontos;
- Outros periódicos internacionais com fator de impacto equivalente a B4 ou B5: 2 pontos;
- Conferências internacionais com fator de impacto equivalente a B3 ou superior: 5 pontos;
- Conferências nacionais com fator de impacto equivalente a B3 ou superior, 2 pontos.

* Obs.: para evitar vinculação direta com o Qualis, foi estabelecida para o triênio 2013-2015 a noção de equivalência do fator de impacto, permitindo que usemos os índices de impacto JCR e h-index do Scopus e do Google. Desta forma, o critério de credenciamento para o triênio fica independente das mudanças que podem ocorrer no Qualis. Adicionalmente, o critério é aplicável a qualquer periódico e evento, independentemente de constar nas listas do Qualis. Esta equivalência deverá ser revisada no final do triênio 2013-2015 para aplicação no triênio subsequente. O Apêndice 1 (Critérios de Credenciamento e Classificação) traz detalhes.

3.2 Fluxo de Trabalho no C2D

Considerando que dados sobre membros do PPGI, suas publicações, respectivos fatores de impacto e classificação Qualis, etc. encontram-se já registrados em outros sistemas e documentos, o C2D deve funcionar de maneira mais automatizada possível, obtendo estes dados e armazenando em sua base de dados local, atualizando-os sempre que necessário. Então, temos o seguinte fluxo de trabalho para o C2D:

1. Na instalação do sistema, cadastra-se a secretária do programa, seu nome, e-mail, cpf e senha para que possa administrar o sistema;
2. A secretária pede ao C2D que cadastre os docentes do PPGI, extraindo suas informações (nome completo, tipo de membro e link para Currículo Lattes) da página de membros do PPGI: <http://informatica.ufes.br/pos-graduacao/PPGI/lista-de>

docentes (note que para extrair o link é necessário clicar no nome do docente e abrir sua página de detalhes);

3. A secretária cadastra manualmente os diferentes níveis de classificação do Qualis (A1, A2, B1, B2, B3, etc.), o sistema de pontuação vigente e os requisitos de publicação vigentes, podendo cadastrar quantos requisitos quiser (atualmente o PPGI usa dois, como descrito anteriormente). Para cada requisito, além do nome e da vigência (data de início e fim), é necessário associar a cada nível de classificação uma pontuação nacional e uma internacional;
4. A secretária pede ao C2D que extraia informações de classificação Qualis de conferências e periódicos. Atualmente, a classificação de conferências pode ser encontrada em uma tabela no documento PDF disponível em — Já a classificação de periódicos encontra-se no sistema Sucupira, seguindo as instruções disponíveis no seguinte documento PDF: —

Note que: nem o PDF das conferências, nem o sistema Sucupira indicam se o veículo é nacional ou internacional, portanto tal informação deve ser completada manualmente;
5. A última extração de dados que a secretária pode pedir ao C2D para fazer é recuperar as publicações dos docentes a partir de seus Currículos Lattes;
6. Com os docentes, critérios, qualificação e publicações cadastrados (atualizados), a secretária finalmente solicita ao C2D que calcule a pontuação de publicação de cada docente e informa, para cada um deles, se atendem aos requisitos cadastrados (no caso do PPGI, os dois requisitos que são avaliados atualmente).

3.3 Atualização de Dados

O cadastro de docentes (passo 2 acima) pode ser repetido de tempos em tempos, quando a secretária assim desejar. Neste caso, espera-se que o C2D faça uma atualização semi-automática, mostrando os docentes atualmente cadastrados, docentes a manter como estão (não muda nada), docentes que mudaram de tipo, novos docentes a cadastrar (presentes na lista de docentes do site mas não no cadastro) e docentes a remover do programa (presentes no cadastro mas não na lista de docentes). Neste último caso, o C2D não deve excluir docentes, mas sim indicar que sua participação como membro do PPGI se encerrou em uma determinada data.

O cadastro da classificação Qualis (passo 4) também deve ser repetido de tempos em tempos, fazendo uma atualização semi-automática similar à descrita para os docentes. É importante ressaltar que o veículo deve ser cadastrado uma só vez e ter associado a ele múltiplas classificações, cada uma com seu(s) ano(s) de vigência.

A extração de dados do Lattes (passo 5) é limitada pelo fato do mesmo utilizar captcha para impedir a obtenção automática dos currículos. O C2D deve tentar uma obtenção semi-automática neste caso. Em último caso, pode-se pedir que a secretária obtenha o currículo em formato XML manualmente e faça o upload do mesmo no sistema. Além disso, os dados de publicação nos currículos são informados pelos próprios docentes e não há garantia que o nome dos veículos cadastrados por eles em seus currículos bate com o nome dos veículos obtidos na lista de conferências / periódicos do Qualis. O C2D deve tentar fazer o máximo possível para reconhecer automaticamente os veículos de cada publicação, utilizando a abordagem semi-automática sempre que necessário.

3.4 Dados interligados

O C2D deve publicar os dados de sua base de dados em formatos de dados interligados de acordo com o W3C (RDF/OWL). Além disso, deve ser avaliada a possibilidade de obter informações adicionais em outras bases de dados interligados existentes, caso seja de utilidade ao sistema de alguma forma (Ex.: dados de conferências na base do DBLP).

4 Requisitos de Usuário

Utilizando das informações citadas anteriormente, foram identificados os seguintes requisitos de usuário e regras de negócio:

Tabela 1 – Requisitos Funcionais

ID	Descrição	Prioridade	Depende
RF-1	O sistema deve cadastrar docentes.	Alta	-
RF-2	O sistema deve cadastrar e autenticar os usuários.	Alta	RF-1
RF-3	O sistema deve cadastrar critérios para avaliação docente.	Média	-
RF-4	O sistema deve calcular pontuação de docentes para entrada no PPGI.	Alta	RF-3, RF-1
RF-5	O sistema deve cadastrar publicações.	Alta	RF-3
RF-6	O sistema deve extrair qualificação do Qualis.	Alta	-
RF-7	O sistema deve cadastrar níveis do Qualis.	Alta	RF-6
RF-8	O sistema deve publicar os dados em formato de dados interligados.	Alta	-

Tabela 2 – Regras de Negócio

ID	Descrição	Prioridade	Depende
RN-1	Para ser classificado / se manter como membro do PPGI e/ou orientador de doutorado, um docente ele deve ter uma pontuação mínima exigida pelo programa (em critério específico).	Alta	RF-1, RF-4
RN-2	O docente deve ter vínculo funcional-administrativo com a instituição.	Média	RF-1

Tabela 3 – Requisitos Não Funcionais

ID	Descrição	Categoria	Escopo	Prioridade
RNF-1	O sistema deve prover o controle de acesso às suas funcionalidades de acordo com as restrições de cada usuário.	Segurança e Acesso	Sistema	Alta
RNF-2	O sistema deve ser de aprendizado fácil, intuitivo, não sendo necessário nenhum treinamento especial para seu uso.	Facilidade de Aprendizagem	Sistema	Média

RNF-3	A ferramenta deve estar disponível como uma aplicação Web, acessível a partir dos principais navegadores disponíveis no mercado.	Portabilidade	Sistem	Média
RNF-4	O sistema deve ser de fácil operação, não sendo necessário uso contínuo para uma boa operação do sistema.	Facilidade de Operação	Sistema	Alta
RNF-5	O sistema deve ser fácil de manter, de modo a acomodar novas funcionalidades ou até mesmo adaptação para organizações específicas.	Facilidade de Manutenção	Sistema	Alta
RNF-6	O desenvolvimento do sistema deve explorar o potencial de reutilização de componentes, tanto no que se refere ao desenvolvimento com reúso quanto ao desenvolvimento para reúso.	Reusabilidade	Sistema	Alta

5 Modelo de Casos de Uso

6 Modelo Estrutural

7 Dicionário de Projeto

Apêndices

APÊNDICE A – Critérios de Credenciamento e Classificação

Este documento apresenta os critérios de credenciamento e de classificação de docentes do PPGI, assim como os requisitos mínimos de publicação aprovados em dezembro de 2012 para o triênio 2013-2015.

1. Ingresso no programa (em fluxo contínuo)

- a) Ter plano de trabalho aprovado pelo colegiado, considerando alinhamento com as linhas de pesquisa do programa.
- b) Atender aos requisitos mínimos de publicação para permanência no programa considerando-se os trabalhos publicados no triênio anterior. Será considerado colaborador até que atenda aos requisitos para ser categorizado como permanente.

2. Critério de credenciamento e recredenciamento no programa (aplicado anualmente a partir de jan/2013):

- a) Atender aos requisitos mínimos de publicação para permanência no programa considerando-se os trabalhos publicados no triênio anterior ou ser bolsista de produtividade CNPq. Caso o docente não atenda a estes requisitos não poderá iniciar novas orientações.
 - Se for docente colaborador, continuará nesta condição enquanto houver alunos sob sua orientação, ou, então, não havendo alunos sob sua orientação, por um ano se manifestar seu interesse. Será descredenciado se não atender aos requisitos mínimos de publicação (item a) após esse período;
 - Se for docente permanente, passará à condição de colaborador enquanto houver alunos sob sua orientação, por um ano se manifestar seu interesse. Será descredenciado se não atender aos requisitos mínimos de publicação (item a) após este período. [Poderá voltar à condição de docente permanente se, quando foram avaliados os critérios de categorização, atender aos requisitos para ser categorizado como permanente e se atender aos requisitos mínimos de publicação (item a).]

3. Critério de categorização (a ser aplicado anualmente a partir de jan/2016):

Permanente e o docente que:

- a) Lecionou ao menos duas disciplinas no programa no triênio anterior (excetuando-se estudos dirigidos);
- b) Concluiu ao menos uma orientação no programa no triênio anterior (excetuando-se co-orientação);
- c) Tem vínculo funcional-administrativo com a instituição;

Os outros docentes são colaboradores ou visitantes.

Obs.: Este critério operacionaliza o Art. 2º da Portaria 191 da CAPES. O item “c” é oriundo desta portaria.

4. Critério para credenciamento como orientador no doutorado (aplicado anualmente a partir de jan/2013):

- a) Ter concluído 2 orientações de mestrado e/ou 1 orientação de doutorado (em qualquer período e programa de pós-graduação stricto sensu).
- b) Atender a requisitos mínimos de publicação no triênio anterior (DIFERENTE DOS REQUISITOS MÍNIMOS DO PROGRAMA)

Obs.: caso um docente que esteja credenciado com orientacoes em andamento nao atenda aos requisitos minimos de publicacao para docentes no doutorado, nao podera iniciar novas orientacoes de doutorado.



Documento de Projeto de Sistema

C2D - Módulo de Credenciamento e Classificação de Docentes do Sistema Marvin

Registro de Alterações:

Versão	Responsável	Data	Alterações

Vitória, ES

2017

1 Introdução

Este documento apresenta o documento de projeto (*design*) do C2D - Módulo de Credenciamento e Classificação de Docentes do Sistema Marvin. Este documento está organizado da seguinte forma: a Seção 2 apresenta a plataforma de software utilizada na implementação da ferramenta; a Seção 3 trata de estratégias utilizadas para tratar requisitos não funcionais (atributos de qualidade); por fim, a Seção 4 apresenta o projeto da arquitetura de software, apresentando cada uma de suas camadas.

2 Plataforma de Desenvolvimento

Na Tabela 1 são listadas as tecnologias utilizadas no desenvolvimento da ferramenta, bem como o propósito de sua utilização.

Tecnologia	Versão	Descrição	Propósito
Java	8.1	Linguagem de programação interpretada, orientada a objetos, compilada para byte-code, interpretada pela Java Virtual Machine(JVM) e de tipagem forte).	Projetada tanto para programação de codificação rápida quanto para programação robusta.
Wildfly	10.1.0	Servidor de aplicação Java EE desenvolvido em Java.	Fornecer ambiente integrado para a execução e gerenciamento eficaz de aplicativos de negócios baseados em servidores. Além disso, auxilia no monitoramento dos serviços disponibilizados.
JavaServerFaces	2.3	Framework web MVC para a construção de interfaces de usuário baseadas em componentes.	Estabelecer um padrão para a construção de interfaces com o usuário do lado do servidor, separando a lógica da aplicação da apresentação e ligando a camada de apresentação ao código do aplicativo.
Facelets	2.0	Sistema <i>opensource</i> de template para aplicações web.	Reutilizar estrutura que é comum às páginas para facilitar manutenção futura do padrão visual. do sistema
PrimeFaces	6.1	biblioteca de componentes de interface gráfica para as aplicações web baseadas em JSF	aumentar a produtividade do desenvolvedor e a experiência do usuário com a aplicação, além de ser flexível e personalizável, com uma grande opção de componentes.
MySQL Server	5.7.20	Sistema Gerenciador de Banco de Dados Relacional.	Persistência dos dados manipulados pela ferramenta.
Git	2.10	Sistema de controle de versão distribuído projetado para lidar com velocidade e eficiência.	Assegurar o controle de versão da aplicação.

Tabela 1 – Plataforma de Desenvolvimento e Tecnologias Utilizadas

Na Tabela 2 vemos os softwares que apoiaram o desenvolvimento de documentos e também do código fonte.

Tecnologia	Versão	Descrição	Propósito
Eclipse Neon	4.6.3	Ambiente de desenvolvimento (IDE) para a linguagem Java.	Facilitar a atividade de implementação de software.
Astah Profissional	7.0.0	Ferramenta para modelagem em UML	Modelar diagramas de classes, casos de uso etc.
TeXstudio	2.12.6	Editor de \LaTeX .	Escrever a documentação do sistema.

Tabela 2 – Softwares de Apoio ao Desenvolvimento do Projeto

3 Atributos de Qualidade e Táticas

Na Tabela 3 são listados os atributos de qualidade considerados neste projeto, com uma indicação se os mesmos são condutores da arquitetura ou não e as táticas a serem utilizadas para tratá-los.

Categoria	Requisitos Não Funcionais	Condutor da Arquitetura	Tática
Portabilidade	RNF-3	Sim	O sistema deve ser organizado segundo o padrão MVC (<i>Model, View, Controller</i>).
Redigibilidade	RNF-7	Sim	Utilizar uma linguagem que seja compatível com os principais navegadores do mercado.
Facilidade de Aprendizado	RNF-4	Sim	O sistema possui uma interface amigável e intuitiva, auxiliando no entendimento do sistema e de suas funcionalidades.
Usabilidade	RNF-6	Não	Prover ao usuário a capacidade de entrar com comandos que permitam operar o sistema de modo mais amigáveis. Para tal, as interfaces do sistema devem permitir, sempre que possível, a entrada por meio de seleção ao invés da digitação de campos.
Confiabilidade	RNF-5	Sim	Sempre que o usuário inserir informações em um formulário, o sistema deve garantir que todos os campos obrigatórios tenham sido devidamente preenchidos, evitando qualquer tipo de inconsistência.
Segurança	RNF-2	Não	O sistema deverá alertar ou redirecionar o usuário quando este tentar realizar uma operação que não esteja liberada para seu acesso.
Confidencialidade	RNF-3	Sim	O sistema deverá aplicar uma função de dispersão utilizando o algoritmo MD5 em informações sensíveis a fim de garantir o sigilo dos dados.

Tabela 3 – Atributos de Qualidade e Táticas Utilizadas

4 Arquitetura de Software

A arquitetura de software do módulo C2D baseia-se no padrão MVC (*Model*, *View*, *Controller*) representado na Figura 1. s

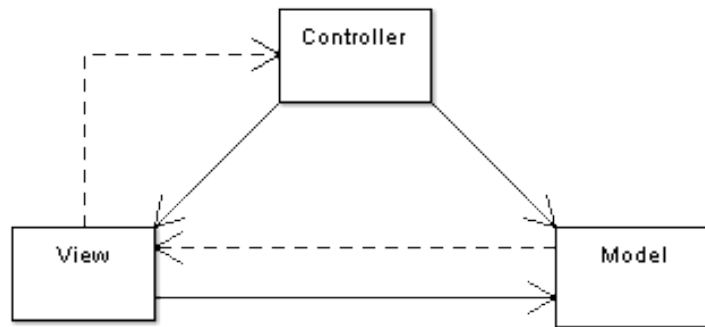


Figura 1 – Representação do padrão de design MVC.

Nas próximas seções, serão apresentados diagramas no padrão proposto pelo *FrameWeb* (SOUZA, 2007).

4.1 Camada *Model*

A seguir, é apresentado o **Modelo de Entidade** seguindo o método *FrameWeb*.

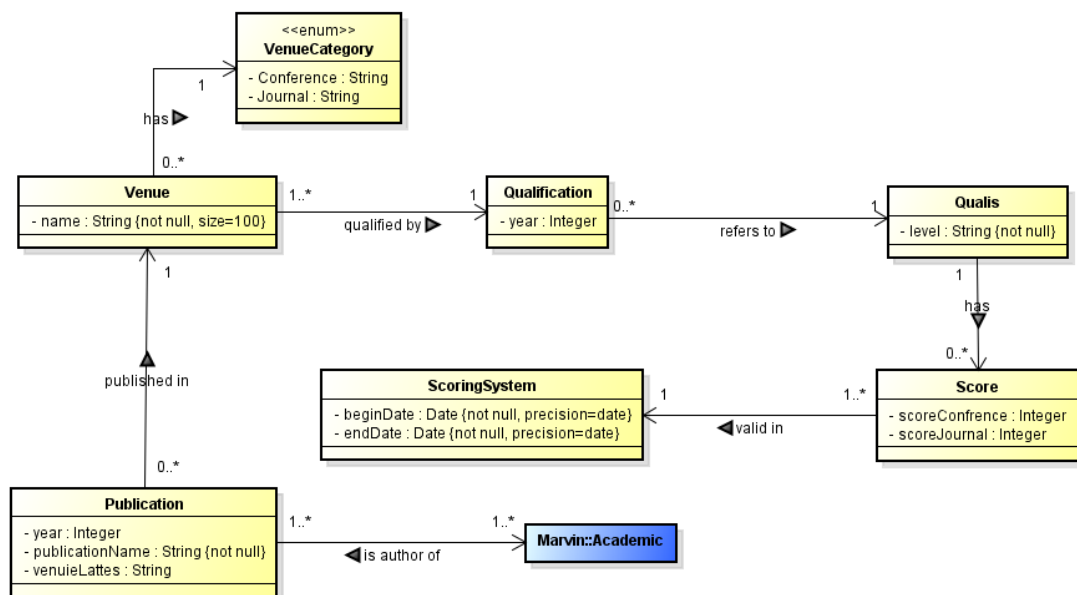


Figura 2 – Modelo de Entidades do módulo C2D.

4.2 Camada View e Controller

As funcionalidades criar, visualizar, editar e excluir (abreviadas de CRUD, do inglês *create, read, update and delete*), seguem um mesmo fluxo de execução e de interação com o usuário. Para todo caso de uso cadastral, essas funcionalidades são semelhantes, utilizando do fato de que todas as classes de domínio estendem de `CrudService`.

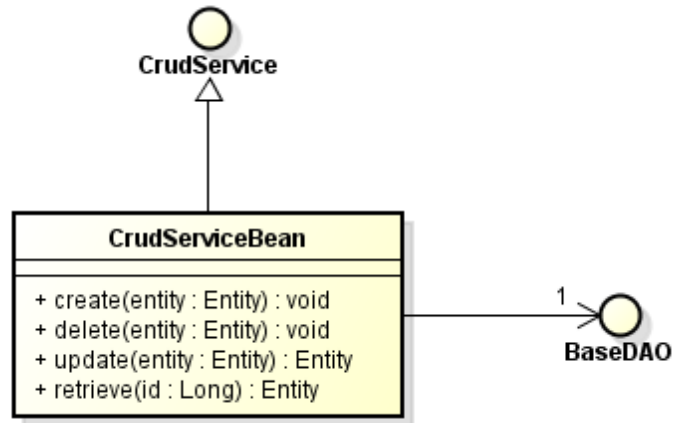


Figura 3 – Classe `CrudService` e métodos CRUD.

Nas Figuras 4 e 5, é apresentado o **Modelo de Aplicação** do sistema C2D.

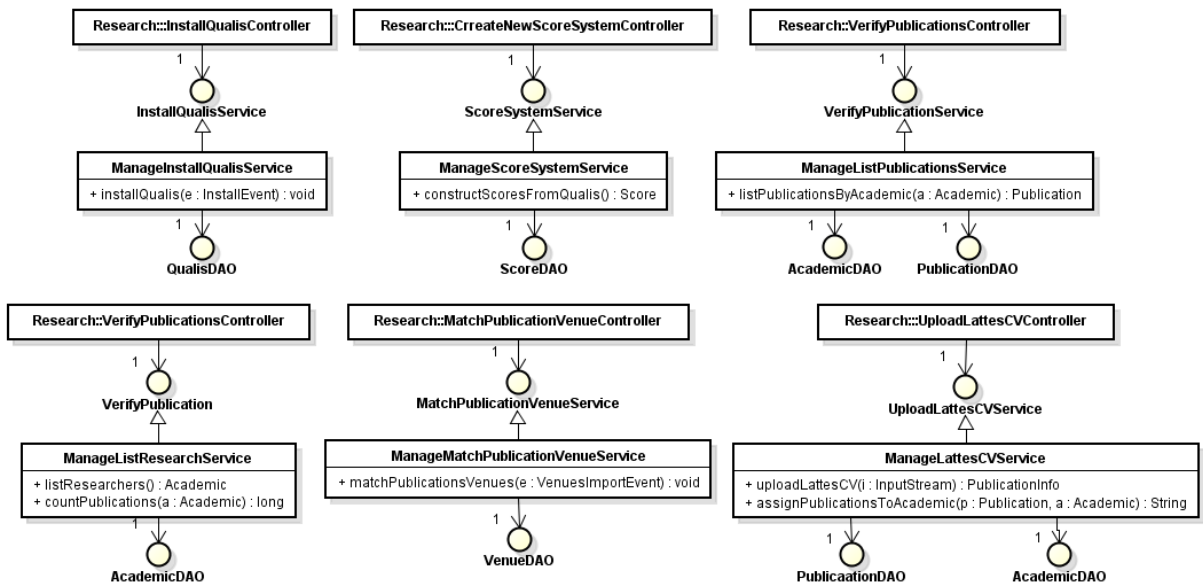


Figura 4 – a - Modelo de Aplicação do módulo C2D.

A Figura 6 apresenta o **Modelo de Navegação** genérico seguido por todas as entidades que são do tipo CRUD, que herdam de `CrudService`.

Essa página possui um formulário relacionado, `manageEntityList`, que armazena todas as entidades sendo mostradas na página citada anteriormente. Ao selecionar uma entidade da tabela, a mesma é relacionada ao atributo `entity` existente em

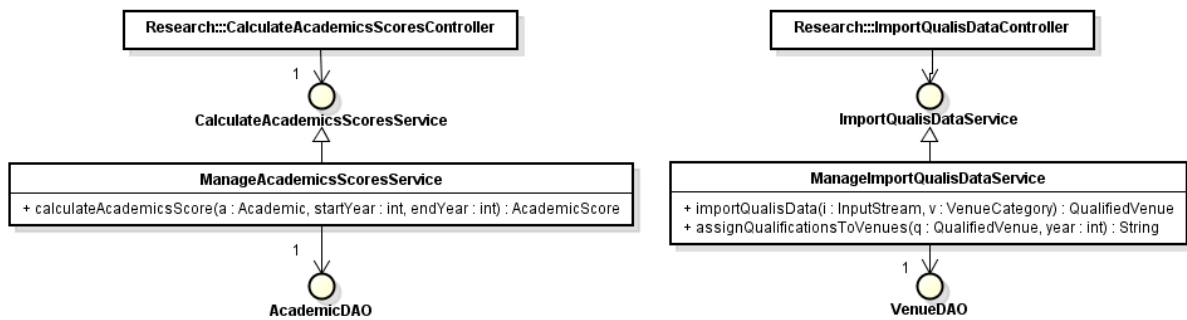


Figura 5 – b - Modelo de Aplicação do sistema C2D.

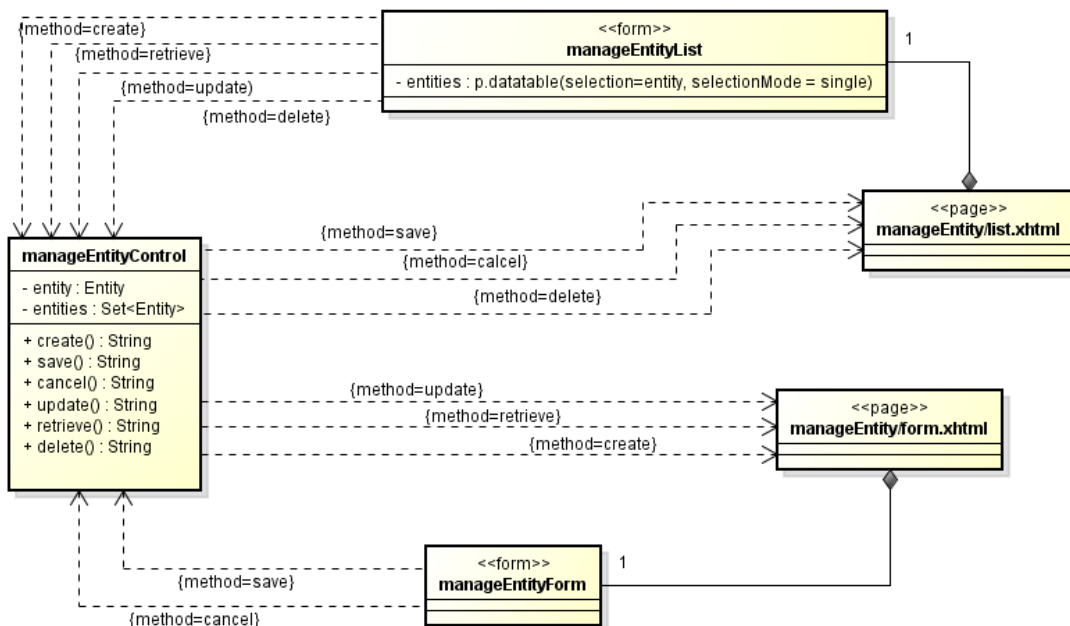


Figura 6 – b - Modelo de Navegação CRUD.

ManageEntityControl, possibilitando executar ações como: atualizar o objeto selecionado (*update*), ler o objeto selecionado (*retrieve*), e excluir o objeto selecionado (*delete*) e até mesmo criar um novo objeto (*create*). Todas estas funcionalidades, ao serem executadas, irão chamar os respectivos métodos existentes em ManageEntityControl. Ao executar tais métodos, existem certos retornos, identificados nas relações de dependência saindo de ManageEntityControl. Como exemplo, quando o usuário executa um *create*, o ele é redirecionado para a página *form.xhtml*, onde insere os dados do novo objeto, e então executa o método *save*, salvando o objeto, e como retorno, irá ser redirecionado para a página inicial (contendo a listagem de todas as entidades existentes no banco).

Para os demais casos de uso, o modelo de navegação mostrado anteriormente não pode ser aplicado, pois possuem métodos diferentes das funções básicas de cadastro. A seguir são representados os modelos de navegação para outros casos de uso do sistema. A Figura 7 representa o modelo de navegação para o caso de uso Calcular Pontuação de Docente e a Figura 8 representa o modelo de navegação para o caso de uso Extrair

Informação de Qualificação

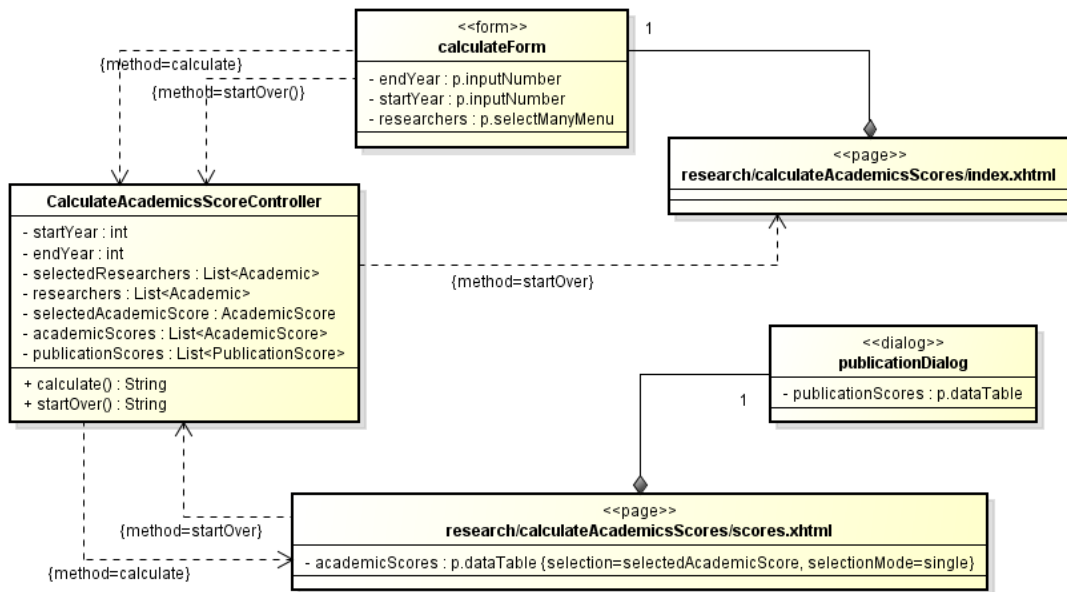


Figura 7 – b - Modelo de Navegação CalculateAcademicScore.

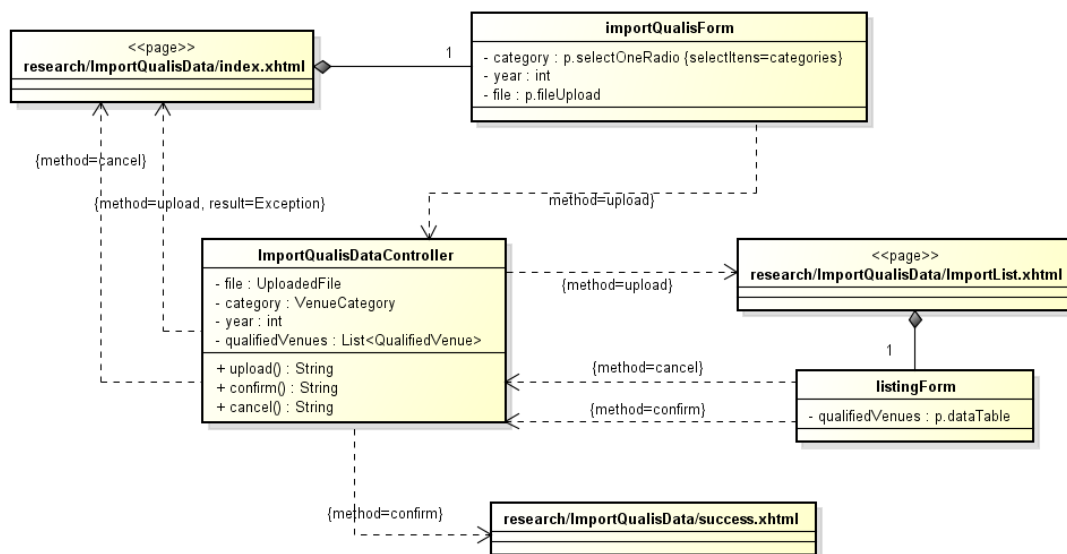


Figura 8 – b - Modelo de Navegação ImportQualisData.

Por fim, temos o **Modelo de Persistência** é um diagrama de classes da UML que representa as classes DAO existentes, responsáveis pela persistência das instâncias das classes de domínio.

Para que não haja repetição de funcionalidades como salvar objetos, retornar todos os objetos, atualizar etc, todas as classes DAO existentes herdam as propriedades da classe *BaseDAO*, apresentada na Figura 9

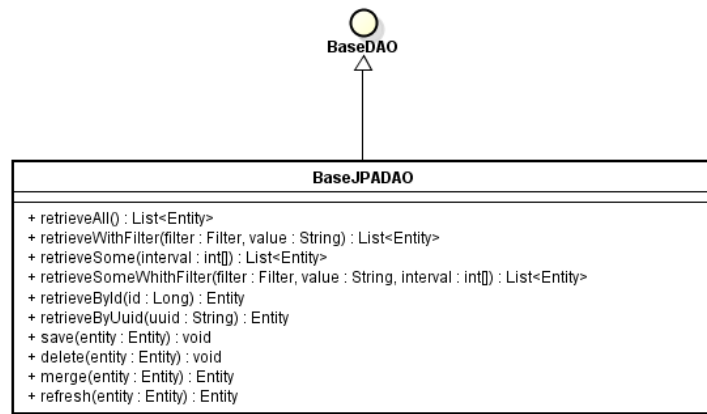


Figura 9 – Classe *BaseDAO* e principais métodos.

A Figura 10 apresenta as classes DAO's existentes no sistema C2D.

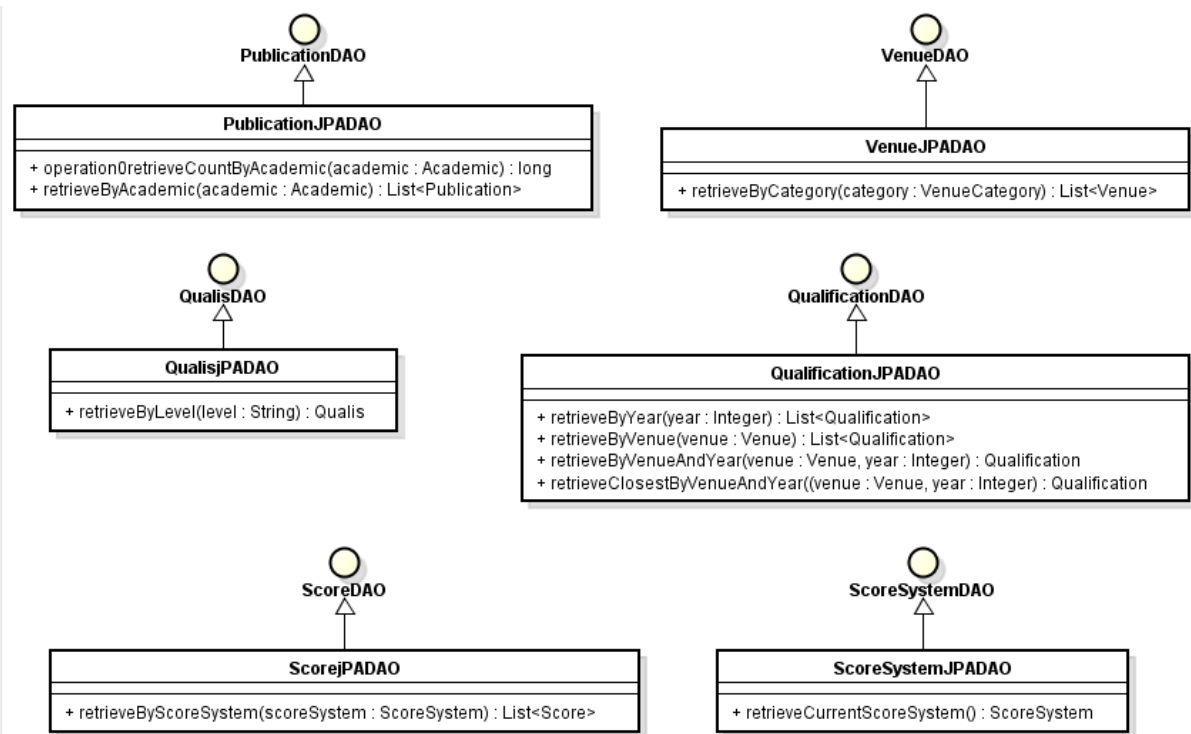


Figura 10 – Modelo de Persistência do módulo C2D.

Referências

SOUZA, V. E. S. *FrameWeb – A Framework-based Design Method for Web Engineering*. Dissertação (Mestrado) — Universidade Federal do Espírito Santo, 2007. Citado na página 5.