

UNIVERSIDADE FEDERAL FLUMINENSE  
INSTITUTO DE COMPUTAÇÃO

RODRIGO DIAS FERREIRA  
TIAGO MANUEL PADRELA AMARO

SAPOS - SISTEMA DE APOIO À PÓS-GRADUAÇÃO

Niterói  
2013

RODRIGO DIAS FERREIRA  
TIAGO MANUEL PADRELA FERREIRA AMARO

SAPOS - SISTEMA DE APOIO A PÓS-GRADUAÇÃO

Trabalho de Conclusão de Curso Apresentado  
ao Curso de Graduação em Ciência da  
Computação da Universidade Federal  
Fluminense para obtenção do Grau de  
Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Leonardo Gresta Paulino Murta  
Coorientadora: Prof. Dr. Vanessa Braganholo Murta

Niterói  
2013

RODRIGO DIAS FERREIRA  
TIAGO MANUEL PADRELA FERREIRA AMARO

SISTEMA DE APOIO A PÓS-GRADUAÇÃO

Trabalho de Conclusão de Curso Apresentado  
ao Curso de Graduação em Ciência da  
Computação da Universidade Federal  
Fluminense para obtenção do Grau de  
Bacharel em Ciência da Computação.

Aprovada em Março de 2013.

BANCA EXAMINADORA

---

Prof. Dr. LEONARDO G. PAULINO MURTA – Orientador  
UFF

---

Prof. Dr. VANESSA BRAGANHOLO MURTA – Orientador  
UFF

---

Prof. Dr. ALEXANDRE PLASTINO DE CARVALHO  
UFF

---

Prof. Dr. CELSO DA CRUZ CARNEIRO RIBEIRO  
UFF

Niterói  
2013

Dedicamos este trabalho a Coordenação da Pós-Graduação em Ciência da Computação da Universidade Federal Fluminense e aos nossos orientadores Leonardo Murta e Vanessa Braganholo. Que o resultado desse esforço possa gerar muitos frutos.

## AGRADECIMENTOS (RODRIGO)

Aos meus pais Antônio Ferreira e Dalila, e à minha irmã Juliana, que estiveram comigo em todos os momentos dessa caminhada, me apoiando nos momentos de dificuldade e comemorando as vitórias sempre ao meu lado. Gostaria de agradecer especialmente à minha madrinha Luiza Dias que, junto com minha mãe, fizeram com que eu saísse de Nilópolis. Sem vocês eu não estaria onde estou hoje.

Aos meus avós Manuel Xavier (*in memoriam*) e Maria Isaura (*in memoriam*) pela educação, conhecimento e carinho durante mais de vinte anos. Espero que, onde quer que estejam, a conclusão desse curso traga um pouco de orgulho para vocês.

Aos demais parentes por compreenderem a minha ausência nos eventos de família e aos amigos por estarem presentes nos momentos de felicidade e tristeza. Em especial gostaria de agradecer à Higor Ortiz e Rogério Lucas pela amizade, conselhos e ensinamentos. Vocês foram e são dois irmãos pra mim.

Ao amigo Tiago Amaro que aceitou o desafio de desenvolver o SAPOS. A sua capacidade técnica e habilidade de aprender rapidamente um novo assunto trouxeram qualidade ao produto que foi gerado.

À minha namorada Carol Cruz (O Oráculo) pela paciência sem tamanho nas discussões sobre a monografia, pelas inúmeras revisões de texto e pelo apoio nos momentos de desespero. Você faz parte desta monografia tanto quanto qualquer um de nós.

Aos orientadores Leonardo Murta e Vanessa Braganholo pela paciência, tempo e dedicação para com este trabalho. Posso dizer que a dedicação de ambos aos seus alunos é um diferencial, digo isso não só como aluno, mas também como amigo.

À Coordenação e Secretaria da Pós-Graduação em Ciência da Computação. Em especial aos Professores Celso Ribeiro e Simone Martins e às secretárias Viviane e Teresa pelo tempo disponibilizado e pela ajuda no desenvolvimento de um sistema cada vez melhor.

Aos docentes da UFF pelo conhecimento passado. Em especial aos professores Leonardo Murta, Alexandre Plastino, Petrucio Viana e Rodrigo Salvador pela dedicação diferenciada aos seus alunos e cuidado na formulação das aulas. Vocês fizeram e fazem a diferença.

A todos os integrantes do Laboratório Tempo, em especial ao professor Julius Leite pela oportunidade de realizar iniciação científica, vivenciando os desafios da pesquisa. Um agradecimento especial aos amigos Giulio Bottari, Matheus Erthal, Douglas Mareli, Carlos

Sant'ana e Vinícius Petrucci que estiveram comigo nos quase dois anos e meio de laboratório. A amizade de vocês foi o maior bem com o qual fui presenteado nesse período.

Aos amigos do Ministério Público do Trabalho, onde realizei meu primeiro estágio e permaneci por um ano. O conhecimento passado vai de noções de programação em PHP até conceitos de Direito Constitucional. No entanto, o mais importante foram os amigos e as experiências vividas nesse período. Por isso, eu agradeço.

Aos amigos da Bridge Consulting. Posso dizer que nunca vi um lugar que reúne tantas pessoas brilhantes e extremamente competentes. Na Bridge aprendi (e ainda estou aprendendo) que o conhecimento nunca é demais e que, mesmo no mercado de trabalho, a pesquisa é um diferencial importante.

Por fim, gostaria de agradecer à UFF, por me proporcionar tanto aprendizado e experiência.

## AGRADECIMENTOS (TIAGO)

À minha mãe Adelaide Amaro, que sempre esteve ao meu lado em todos os momentos.

À Bia Ramalho, companheira e amiga, na alegria e na tristeza.

Ao meu irmão Carlos Amaro Júnior, amigo e um pouco de pai.

Aos meus amigos e conselheiros: Caio Siqueira, Luiz Felipe Gonçalves (Fausto), Felipe Pinho, Marcela Rockenbach, Renato Guzzardi e Yuri Gaidarji.

À Juliana Affonso, que dedicou de seu tempo e arte para o novo design do SAPOS.

À Carol Cruz, nosso oráculo e consultora.

Ao Rodrigo Dias, por me convidar no início da construção deste magnífico projeto e por toda sua paciência por não ter se arrependido de tal ato.

A todo corpo docente e à coordenação de Pós-Graduação em Ciência da Computação da Universidade Federal Fluminense, em especial: Celso Ribeiro, Simone Martins e as secretárias Teresa Cancela e Viviane Aceti.

Aos orientadores Leonardo Murta e Vanessa Braganholo, que executam com maestria, dedicação e excelência todos seus trabalhos acadêmicos. Agradeço profundamente a vocês a paciência e o aprendizado obtido com este projeto.

Agradeço também à Universidade Federal Fluminense, que tanto me ensinou e educou para a vida profissional e pessoal.

*“Imagination is more important than knowledge. For knowledge is limited, whereas imagination embraces the entire world, stimulating progress, giving birth to evolution.”*

– Albert Einstein

*“The purpose of abstraction is not to be vague, but to create a new semantic level in which one can be absolutely precise”*

– Edsger Dijkstra

*“The more you know, the more you realize you know nothing.”*

– Sócrates



## RESUMO

A Pós-Graduação em Computação da Universidade Federal Fluminense utiliza uma variedade de sistemas para o gerenciamento de dados referentes a alunos, professores e bolsas de fomento, o que muitas vezes gera informações duplicadas e inconsistências. Além disso, esses sistemas não atendem todas as necessidades dos usuários responsáveis pela administração, e muitas informações são controladas manualmente, em planilhas e documentos diversos. A proposta deste trabalho é o desenvolvimento de uma aplicação *web* que visa cobrir as lacunas deixadas pelos sistemas em utilização. O novo sistema foi denominado SAPOS (Sistema de Apoio à Pós-Graduação). O projeto visa melhorar a gestão de informações já existentes em sistemas legados e a inserção de novas funcionalidades, como o controle de alocação de bolsas de fomento e o controle de etapas de um aluno. A implementação da proposta foi realizada utilizando uma metodologia de desenvolvimento ágil, que possui como principal característica uma forte interação com o usuário. Esta, por sua vez, diminui a probabilidade da adição de requisitos desnecessários, potencializando o aumento da satisfação do usuário. A cada iteração, uma versão do SAPOS era entregue aos usuários, que puderam ir usando versões preliminares enquanto novas funcionalidades eram implementadas. Foi utilizado o framework Ruby on Rails, que facilita a implementação, manutenibilidade e escalabilidade do sistema. Essas características permitem que o sistema seja facilmente continuado por outros desenvolvedores. A versão desenvolvida nesse trabalho contempla o cadastro de informações de alunos, matrículas, professores, orientações, bolsas e etapas, e vem sendo usada com sucesso pela secretaria e coordenação do Programa de Pós-Graduação em Computação da UFF.

Palavras-chave: Aplicação *Web*, Engenharia de Software, Desenvolvimento Ágil, Ruby on Rails, Gestão Acadêmica de Pós-Graduação.

## ABSTRACT

The Graduate Program in Computer Science of the Fluminense Federal University uses a variety of systems for managing data from students, professors and scholarships, which often creates duplicated information and inconsistencies. In addition, these systems do not meet all the needs of the users responsible for the administration. Much of the information is manually managed in spreadsheets and documents in general. This work's proposal is to develop a *web* application that aims to cover the gaps left by the current systems. The new system was called SAPOS. The project purpose is to improve the management of existing information in legacy systems and the inclusion of new features, such as controlling scholarship allocation and phases of a student. We used an agile development methodology, which has as main characteristic a strong interaction with the user. This methodology decreases the likelihood of adding unnecessary requirements, increasing user satisfaction. In each interaction, a new version of SAPOS was delivered to the users. This way, the system began to be used while new functionalities were being developed. We used Ruby on Rails to implement SAPOS. This allow for easy development, maintainability and scalability of the system. This also allows the development to be easily continued by other developers. The version developed in this work includes management of students, enrollments, professors, advisements, scholarship allocation and phases. SAPOS has been successfully used the coordination of the Graduate Program in Computer Science at UFF.

Keywords: Web Applications, Software Engineering, Agile Development, Ruby on Rails, Post-graduation Academic Management.

## SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS .....	13
CAPÍTULO 1 – INTRODUÇÃO .....	15
CAPÍTULO 2 – TRABALHOS RELACIONADOS .....	18
2.1 SISTEMAS UFF.....	19
2.2 COBALTO .....	21
2.3 SIGPÓS .....	23
2.4 DISCUSSÃO .....	24
CAPÍTULO 3 – SAPOS: SISTEMA DE APOIO A PÓS-GRADUAÇÃO .....	25
3.1 ORGANIZAÇÃO DO SISTEMA .....	25
3.2 ALUNOS .....	26
3.3 PROFESSORES .....	28
3.4 BOLSAS .....	30
3.5 ETAPAS .....	32
3.6 FORMAÇÃO.....	33
3.7 LOCALIDADES .....	34
3.8 CONFIGURAÇÕES.....	34
3.9 CONSIDERAÇÕES FINAIS .....	35
CAPÍTULO 4 – DESENVOLVIMENTO.....	36
4.1 METODOLOGIA ÁGIL .....	36
4.1.1 APLICAÇÃO DE MÉTODOS ÁGEIS NO SAPOS.....	37
4.2 PADRÃO DE PROJETO .....	38
4.2.1 MODELO .....	39
4.2.2 VISÃO .....	39
4.2.3 CONTROLE.....	39
4.2.4 INTERAÇÃO ENTRE AS CAMADAS .....	40
4.3 LINGUAGEM DE PROGRAMAÇÃO.....	40

4.3.1 RUBY .....	40
4.3.2 RUBY ON RAILS .....	41
4.3.3 PRINCIPAIS BIBLIOTECAS UTILIZADAS .....	43
4.4 MIGRAÇÃO DE DADOS .....	44
4.4.1 FERRAMENTA UTILIZADA.....	44
4.5 IMPLEMENTAÇÃO DO SISTEMA.....	45
4.5.1 FERRAMENTAS UTILIZADAS .....	45
4.5.2 HISTÓRICO DE VERSÕES.....	47
4.5.3 DESAFIOS ENCONTRADOS .....	52
4.6 CONSIDREÇÕES FINAIS .....	53
CAPÍTULO 5 – CONCLUSÃO .....	54

## LISTA DE ABREVIATURAS E SIGLAS

CGIC-UFFel: Centro de Gerenciamento de Informações e Concursos da UFPel

COC: *Convention Over Configuration*

CRUD: *Create, Read, Update and Destroy*

DRY: *Don't Repeat Yourself*

IC-UFF: Instituto de Computação da Universidade Federal Fluminense

MVC: Modelo - Visão - Controle

NTi: Núcleo de Tecnologia da Informação

ORM: *Object-Relational Mapping*

PGC: Pós-Graduação em Computação

PREC: Pró-Reitoria de Extensão e Cultura

PRGRH: Pró-Reitoria de Gestão de Recursos Humanos

PROPP: Pró-Reitoria de Pesquisa e Pós-Graduação

RESTFul: *Representational State Transfer*

REUNI: Reestruturação e Expansão das Universidades Federais

SAPOS: Sistema de Apoio à Pós-Graduação

SCPG: Sistema de Controle da Pós-Graduação

SEL: Software Engineering Laboratory

SIAPE: Sistema Integrado de Administração de Recursos Humanos

SIEX: Sistema de Extensão

STi: Superintendência de Tecnologia da Informação

UFF: Universidade Federal Fluminense

UFMS: Universidade Federal de Mato Grosso do Sul

UFPel: Universidade Federal de Pelotas

## LISTA DE ILUSTRAÇÕES

FIGURA 2.1: IDUFF.....	19
FIGURA 2.2: SisPós.....	21
FIGURA 2.3: TELA DE CONSULTA DE CURSOS DO SIGPós.....	23
FIGURA 3.1: TELA DE AUTENTICAÇÃO.....	25
FIGURA 3.2: PRIMEIRA TELA VISUALIZADA DO SISTEMA.....	26
FIGURA 3.3: TELA DA ÁREA DE ALUNOS.....	27
FIGURA 3.4: TELA DA ÁREA DE MATRÍCULAS.....	28
FIGURA 3.5: TELA DA ÁREA DE PROFESSORES.....	29
FIGURA 3.6: TELA DA ÁREA DE ORIENTAÇÕES.....	30
FIGURA 3.7: VALIDAÇÃO DE ALOCAÇÃO DE BOLSAS.....	31
FIGURA 3.8: TELA DE REALIZAÇÃO DE ETAPAS.....	32
FIGURA 3.9: TELA DA ÁREA DE INSTITUIÇÕES E CURSOS.....	33
FIGURA 3.10: TELA DE LOCALIDADES.....	34
FIGURA 3.11: TELA DE CONFIGURAÇÕES.....	34
FIGURA 4.1: UMA VISÃO ABSTRATA DA ARQUITETURA MVC.....	39
FIGURA 4.2: TRECHO DE CÓDIGO QUE IMPRIME CINCO VEZES O TEXTO “ <i>RUBY LANGUAGE</i> ”.....	41
FIGURA 4.3: ARQUITETURA DO RUBY ON RAILS.....	42
FIGURA 4.4: COMANDO DE GERAÇÃO UTILIZADO PELO <i>ACTIVE_SCAFFOLD</i> .....	43
FIGURA 4.5: TRECHO DE CÓDIGO PARA VALIDAÇÃO DE DATAS EM ALOCAÇÃO DE BOLSAS.....	44
FIGURA 4.6: PARTE DA TABELA DE PAÍSES PARA MIGRAÇÃO DE DADOS.....	45
FIGURA 4.7: PRIMEIRA VERSÃO DO MODELO DO SISTEMA.....	47
FIGURA 4.8: MODELO FINAL COMPLETO COM TODOS OS ATRIBUTOS.....	48
FIGURA 4.9: MODELO DE ETAPAS E CONTROLE DE PRORROGAÇÕES.....	49
FIGURA 4.10: BUSCA AVANÇADA EM ORIENTAÇÕES.....	50
FIGURA 4.11: RELATÓRIO DE ORIENTAÇÕES.....	51
FIGURA 4.12: NOVO <i>LAYOUT</i> DO SISTEMA.....	52

## CAPÍTULO 1 – INTRODUÇÃO

O avanço tecnológico tem proporcionado à sociedade uma série de facilidades nos mais diversos setores. Atividades que antes necessitavam de mais tempo e eram feitas de forma manual, hoje estão automatizadas por sistemas computacionais. Dentre as vantagens trazidas pela utilização desses sistemas, pode-se citar (REZENDE, 2005, p. 27): ganho de produtividade no registro e processamento de informações, armazenamento de dados mais eficaz e seguro, maior qualidade das informações registradas, melhor controle no acesso à informação e, além disso, a realização de análises e avaliações mais consistentes, auxiliando na tomada de decisões.

Mesmo diante de tais avanços, a adoção de sistemas automatizados pode ser um processo de longo prazo, devido ao tempo necessário para o desenvolvimento de um sistema. O atraso na adesão de novos métodos e tecnologias interfere no crescimento organizado e eficiente de uma instituição, afetando a sua capacidade de consulta dos dados, gerando inconsistência e acúmulo de informação desordenada (VALENTIM, 2010, p. 249).

De fato, essa era a realidade na secretaria da Pós-Graduação em Computação (PGC) da Universidade Federal Fluminense (UFF), que manipula uma grande quantidade de dados, como: informações sobre alunos, alocação de bolsas, orientações, credenciamento de professores, entre outros. Essas informações eram armazenadas de forma descentralizada, planilhas *Microsoft Excel* e bancos de dados *Microsoft Access*, dificultando a garantia da integridade, consistência e durabilidade dos dados. O crescimento no número de discentes e docentes do PGC contribuiu para agravar o problema, pois como consequência desse aumento, o volume de informações a ser gerenciada também cresceu. De fato, entre os anos de 2006 a 2011, o número de docentes credenciados aumentou de 27 para 38 (BRAGANHOLO, 2013).

Ciente dos problemas citados, o coordenador da PGC, Celso Ribeiro, procurou o Prof. Leonardo Murta, relatando a necessidade de um sistema que centralizasse os dados gerenciados pelo PGC. Assim, nasceu o SAPOS (Sistema de Apoio à Pós-Graduação), produto deste trabalho, que tem como objetivo atender às necessidades e expectativas de gerenciamento de dados da PGC através da substituição dos controles efetuados por planilhas e banco de dados descentralizados. Além disso, a proposta tem o objetivo de inserir funcionalidades que não eram cobertas pelos controles em utilização.

No entanto, para que o objetivo fosse atendido, foi necessário definir o método que seria utilizado no desenvolvimento do sistema. Devido à capacidade de adequar-se a mudanças de requisitos, e a cada iteração gerar versões utilizáveis do sistema, a utilização de métodos ágeis (COHEN; LINDVALL; COSTA, 2004) foi uma escolha natural para o desenvolvimento do SAPOS.

Para que fosse possível a adoção de métodos ágeis, foi necessária a utilização de uma tecnologia condizente com a sua característica iterativa e incremental. Tais características exigiram rapidez e flexibilidade no processo de desenvolvimento e disponibilização do sistema. Assim, decidiu-se por desenvolver um sistema *web*, que, dentre suas diversas características, podem ser destacadas a transparência (para o usuário) na atualização do sistema para novas versões, o acesso via Internet e a capacidade de ser utilizado em qualquer sistema operacional.

Definida a plataforma, o próximo passo foi a escolha da linguagem de programação. Dentre linguagens existentes e amplamente utilizadas no mercado para desenvolvimento web (TIOBE, 2013), o Ruby possui características de fácil leitura e aprendizado (FLANAGAN; MATSUMOTO, 2008, p. 17), além de possuir uma comunidade ativa e um rico conjunto de bibliotecas externas. Tais características foram essenciais na escolha do Ruby como linguagem de desenvolvimento do SAPOS. A seleção dessa linguagem também se deu pelo destaque do framework Ruby on Rails (RUBY; THOMAS; HANSSON, 2011), um conjunto de bibliotecas da linguagem Ruby, utilizado para o desenvolvimento de aplicações web. O uso do framework no projeto é explicado em detalhes no Capítulo 4.

No decorrer deste projeto, outras decisões foram tomadas com a ajuda dos futuros usuários do sistema. Essas escolhas encontram-se explicitadas ao longo deste documento.

O restante deste trabalho encontra-se organizado em quatro capítulos além desta introdução. O Capítulo 2 discute trabalhos relacionados à gestão acadêmico-administrativa de Pós-Graduação, apresentando quatro sistemas de gerenciamento de dados acadêmicos. Tais sistemas foram escolhidos por possuírem funcionalidades características a um sistema de gerência de dados administrativos de alunos de Pós-Graduação. O Capítulo 3 introduz o SAPOS, suas telas e funcionalidades. O capítulo apresenta as seções do sistema, onde, em cada uma, são explicitadas as ações que podem ser realizadas e os tipos de dados que são manipulados. O Capítulo 4 mostra o processo de desenvolvimento do sistema e as tecnologias utilizadas. O capítulo apresenta a metodologia de desenvolvimento utilizada no projeto, padrões de projeto aplicados em sua estrutura, a linguagem de programação escolhida e o processo de migração de dados dos antigos sistemas. O capítulo 5 finaliza esta monografia



destacando o diferencial do sistema SAPOS em comparação com os sistemas de outras universidades, a continuidade de seu desenvolvimento e suas limitações.

## CAPÍTULO 2 – TRABALHOS RELACIONADOS

O volume de dados manipulados em uma secretaria de cursos cresce a cada dia. Os incentivos do governo à educação permitem o aumento do número de docentes, de alunos e de bolsas de fomento, fazendo com que a gerência das informações torne-se cada vez mais complexa. Por esse motivo, cresce a demanda por sistemas cujo objetivo seja a centralização e simplificação do acesso à informação.

No meio acadêmico existem sistemas bem estruturados e desenvolvidos, a fim de facilitar cada vez mais o trabalho das pessoas envolvidas com as tarefas de gerência de um curso. Neste capítulo estão descritos três sistemas de gerência de dados em meio acadêmico: (i) Sistemas UFF, compostos pelos sistemas idUFF e SisPós, ambos desenvolvidos pelo antigo NTi (Núcleo de Tecnologia da UFF), atualmente conhecido como STi (Superintendência de Tecnologia da Informação); (ii) Cobalto, desenvolvido pela Universidade Federal de Pelotas (UFPel); e (iii) SigPós, construído pela Universidade Federal de Mato Grosso do Sul (UFMS).

As informações apresentadas ao longo deste capítulo foram obtidas através de uma busca por sistemas de gerência de dados acadêmico-administrativos dos programas de Pós-Graduação das universidades públicas brasileiras. Foi realizada uma verificação nos sites institucionais em busca de informações sobre o tipo de sistema em questão e, quando encontrado um sistema que satisfazia os pré-requisitos descritos, era realizado um contato por e-mail, solicitando as seguintes informações:

- O que é o sistema?
- Quem utiliza o sistema?
- Uma breve descrição das funcionalidades.
- Como surgiu o sistema?
- Antes da criação do sistema, como eram gerenciados os dados da Pós-Graduação?

O e-mail em questão foi enviado para os setores responsáveis pelo desenvolvimento dos sistemas nas Universidades. Entretanto foi necessário o estabelecimento de um prazo para o recebimento de respostas, a fim de não prejudicar a escrita deste trabalho. Assim foi feita a seleção dos sistemas que constam neste capítulo.

Em um segundo contato, foi solicitado aos responsáveis dos sistemas em questão informações mais específicas sobre como eram gerenciados os dados administrativos da Pós-Graduação. Dentre as informações solicitadas, por exemplo, procurou-se saber como eram realizados a gerência das bolsas de fomento, o controle de etapas concluídas por um aluno e o número de prorrogações que poderiam ser solicitadas por um aluno.

## 2.1 SISTEMAS UFF

A UFF possui diversos sistemas para controle da área acadêmica e administrativa. Entretanto, nesta seção são apresentados apenas os sistemas idUFF e o Projeto SisPós. Esses sistemas foram escolhidos por manipular informações referentes à Pós-Graduação, já que são sistemas com o mesmo foco deste trabalho: a Pós-Graduação.

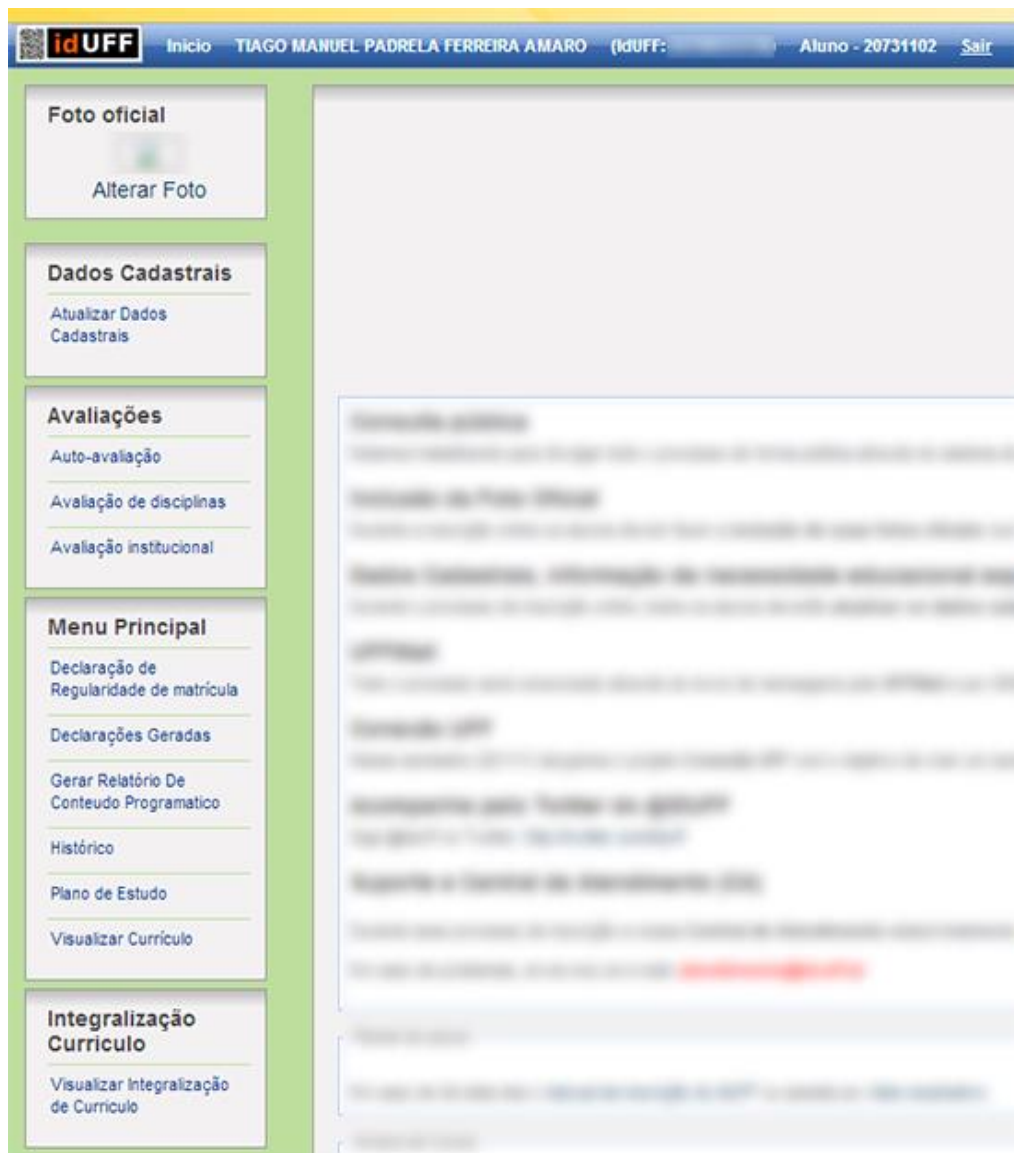


Figura 2.1: idUFF

O idUFF, como mostrado na Figura 2.1, é um sistema de gerência de dados acadêmicos de alunos, ex-alunos, professores e funcionários administrativos da Universidade Federal Fluminense. Dentre os dados acadêmicos a que um aluno tem acesso, pode-se citar: matrícula, coeficiente de rendimento, notas, percentagem do curso concluída, entre outros. Esse sistema tem o objetivo de centralizar informações acadêmico-administrativas dos usuários, permitindo que os mesmos possam visualizar as informações condizentes com seu perfil. Um professor, por exemplo, é capaz de visualizar as turmas de graduação que está ministrando, os alunos que compõem cada turma, efetuar o lançamento de notas, entre outras funcionalidades. Adicionalmente, pode-se destacar a funcionalidade de inscrição em disciplinas, disponível somente para alunos de graduação.

O idUFF tem como foco os alunos de graduação, no entanto os alunos de Pós-Graduação também têm acesso ao sistema. Diferente dos alunos de graduação, os alunos da Pós-Graduação conseguem apenas consultar o número da matrícula. A partir da carência de funcionalidades para alunos da Pós-Graduação, nasceu a iniciativa de desenvolvimento do SisPós, exibido na Figura 2.2.

O Projeto SisPós, ainda em desenvolvimento pelo STi, tem o objetivo de atender às necessidades não contempladas pelo idUFF, permitindo aos funcionários da coordenação: gerenciar um aluno de Pós-Graduação, ou seja, matricular novos alunos, gerenciar editais, emitir carteirinhas, entre outras funcionalidades.

Por mais que o SisPós esteja sendo desenvolvido para preencher as lacunas deixadas pelo idUFF, o foco desse sistema será a gestão acadêmica dos alunos da Pós-Graduação, o que não contempla a gestão administrativa da mesma. O SisPós é focado no cadastro de alunos, deixando carente a gerência de informações administrativas, como o controle de bolsas de fomento, alocação de bolsas a alunos, orientações, entre outras informações importantes para a Pós-Graduação. Para contornar esse problema, cada coordenação de Pós-Graduação da UFF gerencia de sua própria forma os dados de seus alunos.

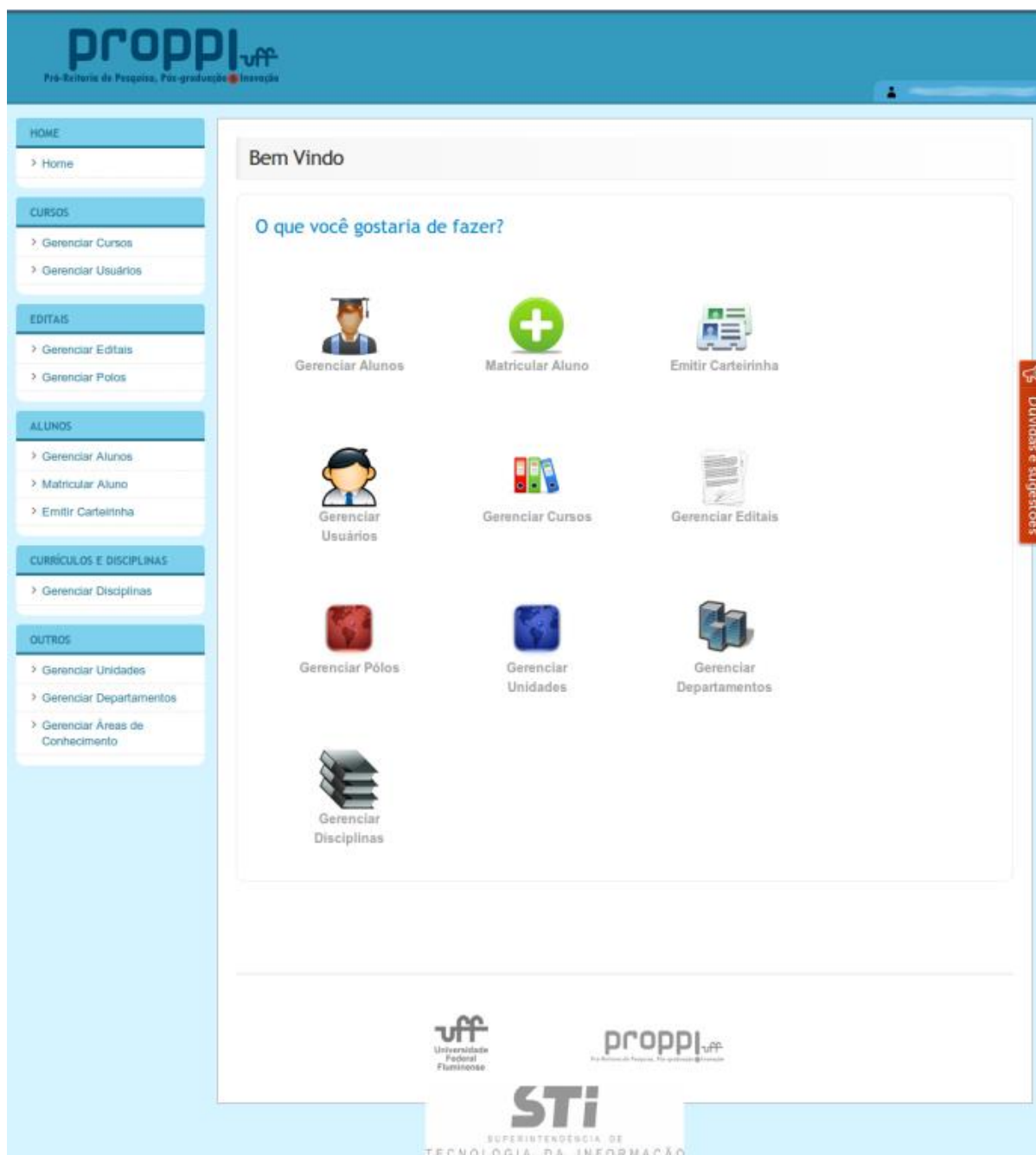


Figura 2.2: SisPós

## 2.2 COBALTO

O Cobalto, batizado conforme o elemento "Co" da tabela periódica, foi desenvolvido pelos técnicos do CGIC-UFPEL (Centro de Gerenciamento de Informações e Concursos da UFPEL) e tem o objetivo de integrar a gerência de dados acadêmicos e administrativos da UFPEL. Em outras palavras, por meio do Cobalto, é possível gerenciar informações referentes ao ensino, pesquisa, extensão e à gestão administrativa.

Segundo AVILA (2012), diretor da área de desenvolvimento de software da UFPEL durante o período de desenvolvimento e implantação do Cobalto, a motivação para a criação

do Cobalto veio da insatisfação e complexidade em gerenciar diversos sistemas desenvolvidos em diferentes plataformas. Diante dessa dificuldade, a universidade sofria regularmente com problemas de inconsistência e duplicação de dados. Para solucionar esses problemas, o projeto Cobalto visou a criação de um ambiente onde novos módulos poderiam ser desenvolvidos compartilhando o mesmo banco de dados.

Dentre os usuários que possuem acesso autenticado ao sistema estão: candidatos a vagas na Universidade, alunos, professores e funcionários administrativos. Para cada tipo de usuário, é liberado o acesso a uma área do sistema, o que permite, por exemplo, que um candidato a uma vaga na Universidade possa: cadastrar os seus dados pessoais, realizar a inscrição, emitir boleto a fim de efetuar o pagamento do concurso a ser realizado, verificar salas das provas, dentre outras funcionalidades.

Diante da urgência em gerenciar os dados dos alunos de forma consistente, o primeiro módulo desenvolvido foi o de Gestão Acadêmica. Ainda segundo AVILA (2012), a Pós-Graduação foi escolhida como modelo para criação desse módulo devido ao sistema acadêmico anterior não contemplar a gestão da mesma. Além disso, fatores como a baixa complexidade de gestão e o menor número de alunos também influenciaram na escolha da Pós-Graduação como modelo para o módulo de Gestão de Ensino.

É importante destacar que o projeto ainda está em desenvolvimento. No momento, o Cobalto possui os seguintes módulos:

**Módulo de Gerenciamento** - Módulo que realiza o controle de acesso dos usuários e de todos os outros componentes do sistema.

**Sistema de Extensão (SIEX)** - Módulo em que os extensionistas conseguem construir e submeter seus projetos. Nesse mesmo portal, a Pró-Reitoria de Extensão e Cultura (PREC) pode obter diversos relatórios de gestão de projetos.

**Sistema de Gerenciamento de RH** - Módulo que envolve o incentivo à qualificação, licença para capacitação, progressão por capacitação e liberação de horário para educação formal. Neste módulo os servidores registram seus requisitos on-line à Pró-Reitoria de Gestão de Recursos Humanos (PRGRH).

**Gestão de Ensino (Graduação e Pós-Graduação)** - Módulo que permite a gestão de notas, ausências, conteúdos, dentre outras informações. Nesse sentido, estudantes e professores interagem com as informações referentes a seus cotidianos acadêmicos.

O módulo de Gestão de Ensino da Pós-Graduação automatizou o registro e processamento das informações acadêmicas (gestão de notas, faltas, conteúdos, entre outras) antes armazenadas em arquivos no *Microsoft Word*, *Microsoft Excel* ou mesmo em meio impresso. No entanto, esse módulo não contempla a gestão de dados administrativos da Pós-Graduação, fazendo com que o controle de bolsas de fomento, controle de prorrogações e a associação entre alunos e professores sejam realizadas de forma descentralizada.

Além dos módulos aqui apresentados, existem outros que estão em desenvolvimento. Dentre eles, destacam-se os portais de pesquisa, mobilidade, egressos, gestão do espaço físico, protocolo e almoxarifado.

## 2.3 SIGPÓS

O SigPós (Figura 2.3), sistema de gestão da Pós-Graduação da Universidade Federal do Mato Grosso do Sul (UFMS), é responsável pelo registro e processamento das informações dos cursos de mestrado, doutorado, especialização e residência.

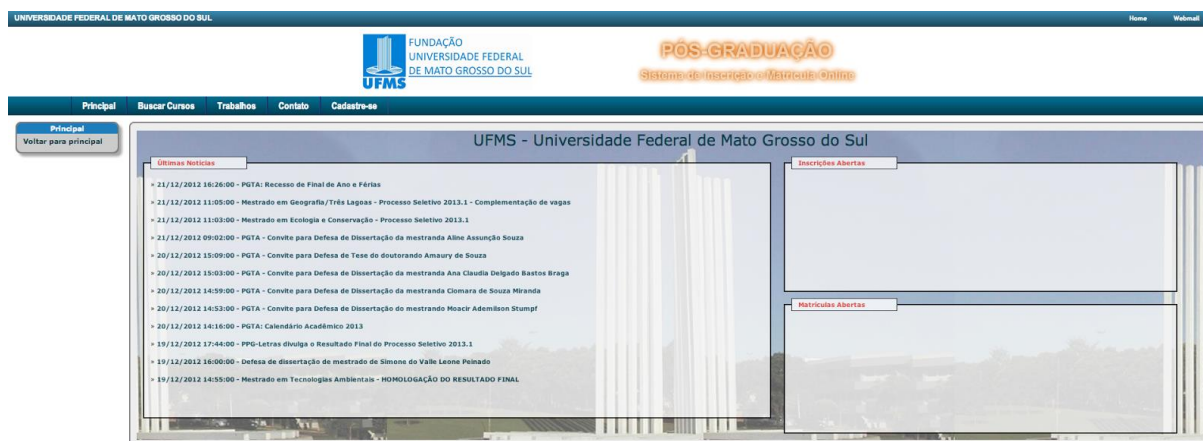


Figura 2.3: Tela de consulta de cursos do SigPós

Dentre o grupo de usuários que possuem acesso ao SigPós, destacam-se: alunos, professores e as secretarias da Pós-Graduação. As secretarias possuem acesso à diversas consultas, como: corpo docente, estrutura curricular, alunos, monografias, dissertações, teses, disciplinas, quadro de disciplinas oferecidas e turmas de ingresso ao programa da Pós-Graduação. Além disso, as secretarias podem gerar o histórico escolar, diplomas, certificados

e outros documentos pertinentes a cada tipo de situação. Os demais usuários têm acesso aos documentos gerados no sistema e à emissão de alguns tipos de relatórios.

Segundo BORGES (2012), assistente de projetos da Coordenadoria de Pós-Graduação/Stricto Sensu da UFMS, a necessidade de implantar melhorias no sistema anterior da UFMS, o SCPG, Sistema de Controle da Pós-Graduação, e o crescimento das informações gerenciadas pela Pós-Graduação foram os principais fatores que motivaram o desenvolvimento do SigPós. Ele foi elaborado a partir de uma equipe composta por profissionais de tecnologia da informação da UFMS e pela equipe do Setor *Stricto Sensu* da PROPP (Pró-Reitoria de Pesquisa e Pós-Graduação). O seu desenvolvimento foi baseado na análise das funcionalidades do SCPG e em sugestões de melhorias enviadas pelas secretarias de cursos.

Diferente dos demais sistemas citados, o SigPós contempla a gestão de dados administrativos da Pós-Graduação, como: controle de bolsas de fomento, alocação de bolsas a alunos, controle de prorrogações, controle de etapas e controle de orientações, onde o professor orientador é informado no cadastro acadêmico do aluno sob sua orientação.

## **2.4 DISCUSSÃO**

Ao longo do capítulo foram apresentados sistemas que lidam com a gestão de informações administrativas em instituições acadêmicas, sendo descritos os principais dados e necessidades que levaram ao desenvolvimento e adoção destes sistemas. Dentre as motivações apresentadas, vale destacar que a organização dos dados foi um objetivo presente em todas as iniciativas de desenvolvimento.

Dentre os sistemas apresentados, todos contemplam a gestão de dados acadêmicos. Quanto às informações administrativas, o SigPós é o único sistema que contempla o seu gerenciamento, sendo ele o sistema mais abrangente quanto à gerência de dados acadêmico-administrativos da Pós-Graduação aqui apresentado.

Em contato por e-mail, os responsáveis pelos demais sistemas demonstraram conhecimento da importância de um sistema para gerenciar os dados administrativos. Vale ressaltar que todas as instituições que fizeram parte desta pesquisa possuem iniciativas de desenvolvimento de módulos que serão responsáveis pelo gerenciamento dos dados em questão.



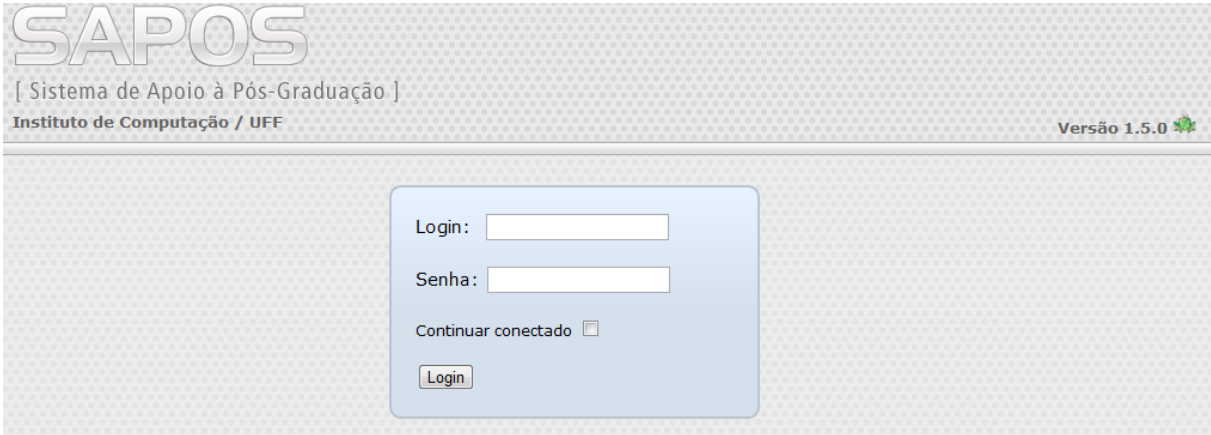
## CAPÍTULO 3 – SAPOS: SISTEMA DE APOIO A PÓS-GRADUAÇÃO

O SAPOS, Sistema de Apoio a Pós-Graduação, tem como proposta central facilitar a gestão de informações referentes à Pós-Graduação. A consulta e atualização de dados tornaram-se tarefas mais simples, devido à automatização do processo cadastral de informações em um único sistema, substituindo a utilização de bases de dados desenvolvidas no *Microsoft Excel* e *Access*.

Neste capítulo é apresentada a forma como o SAPOS está organizado, com uma descrição de cada tela e suas funcionalidades. A Seção 3.1 apresenta a interface do sistema. Na Seção 3.2 estão as informações referentes à vida acadêmica do aluno. A Seção 3.3 descreve as informações básicas sobre professores e seu relacionamento com matrículas. A Seção 3.4 trata do gerenciamento de bolsas de fomento. Na Seção 3.5 é discutido o controle de etapas realizadas por um aluno e os pedidos de prorrogação solicitados pelo mesmo. Na Seção 3.6, é discutido o cadastro de instituições e cursos e sua associação com alunos. A Seção 3.7 contempla as informações referentes à localidade. Na Seção 3.8 estão as configurações de acesso ao sistema. Por fim, a Seção 3.9 contém as considerações finais sobre o capítulo.

### 3.1 ORGANIZAÇÃO DO SISTEMA

Ao acessar o endereço do SAPOS, a primeira tela exibida é a de autenticação, como mostra a Figura 3.1. Em um primeiro momento, os usuários são compostos apenas pelo corpo administrativo da Pós-Graduação.



A imagem mostra a interface de autenticação do sistema SAPOS. No topo, há o logotipo "SAPOS" em uma fonte grande e estilizada, seguido pelo texto "[ Sistema de Apoio à Pós-Graduação ]" e "Instituto de Computação / UFF". No canto superior direito, está escrito "Versão 1.5.0" com um ícone de árvore verde. O formulário de login é centralizado e contém os seguintes elementos: um campo de texto rotulado "Login:", um campo de texto rotulado "Senha:", uma opção de caixa de seleção rotulada "Continuar conectado" e um botão "Login" na base.

Figura 3.1: Tela de autenticação

Após a realização do *login*, é apresentada ao usuário a tela inicial do sistema. Como pode ser visto na Figura 3.2, essa tela é composta por uma barra de abas na parte superior, com as abas Alunos, Professores, Bolsas, Etapas, Formação, Localidades e Configurações. Na parte esquerda da tela é possível visualizar um segundo conjunto de abas subordinado ao superior, ou seja, esse conjunto muda conforme a escolhas feitas no menu superior. No centro da página, podem ser visualizados os registros referentes à opção escolhida no conjunto de abas da esquerda.



The screenshot shows the SAPOS system interface. At the top, there is a header with the logo 'SAPOS' and the text '[ Sistema de Apoio à Pós-Graduação ] Instituto de Computação / UFF'. A navigation bar contains tabs for 'Alunos', 'Professores', 'Bolsas', 'Etapas', 'Formação', 'Localidades', and 'Configurações'. Below this, a sub-menu for 'Matrículas' is active, showing a search bar and an 'Adicionar' button. The main content area displays a table with the following data:

	Aluno	Número de Matrícula	Nível	Tipo de Matrícula	Data de Admissão	Desligamento	
Alunos	Charlie Weasley	DH01	Doutorado	Regular	Junho-1995		Editar Excluir Visualizar
Desligamentos	Bill Weasley	DH02	Doutorado	Especial	Março-1995		Editar Excluir Visualizar
	Harry Potter	MH01	Mestrado	Regular	Junho-1998		Editar Excluir Visualizar
Matrículas	Hermione Granger	MH02	Mestrado	Regular	Março-1998		Editar Excluir Visualizar
	Ronald Weasley	MH03	Mestrado	Regular	Março-1998		Editar Excluir Visualizar
Níveis	Draco Malfoy	MH04	Mestrado	Regular	Junho-1998		Editar Excluir Visualizar
	Luna Lovegood	MH05	Mestrado	Especial	Junho-1998		Editar Excluir Visualizar
Razões de Desligamento	7 Registros Encontrados						
Tipos de Matrícula							

Figura 3.2: Primeira tela visualizada do sistema

### 3.2 ALUNOS

A seção de alunos refere-se não só aos alunos que estão atualmente matriculados na Pós-Graduação do Instituto de Computação da UFF, mas também aos que tiveram seus dados cadastrados no banco de dados *Microsoft Access* que era utilizado anteriormente. Esse armazenamento de dados anteriores tem como objetivo manter um controle do histórico dos discentes.

Esta seção foi subdividida em sete itens, sendo eles: Alunos, Matrículas, Desligamentos, Níveis, Razões de Desligamento e Tipos de matrícula.

Na área **Alunos** (exibida na Figura 3.3) são descritas as características do discente, como: nome, CPF, logradouro, entre outras. Como em todas as outras telas do sistema, é possível reordenar a tabela em questão de acordo com o atributo desejado. Na área de edição do Aluno é possível associá-lo a um ou mais cursos de especialização, graduação, mestrado ou doutorado que o discente tenha cursado anteriormente.

SAPOS  
[ Sistema de Apoio à Pós-Graduação ]  
Instituto de Computação / UFF  
Versão 1.5.0

Logout

Alunos Professores Bolsas Etapas Formação Localidades Configurações

Alunos  Buscar

	Nome	CPF	Matrículas	
Alunos	Alice Tolipan	1111111135	-	Editar Excluir Visualizar
Desligamentos	Bill Weasley	1111111201	DH02 - Bill Weasley	Editar Excluir Visualizar
	Charlie Weasley	1111111200	DH01 - Charlie Weasley	Editar Excluir Visualizar
Matrículas	Cho Chang	1111111131	-	Editar Excluir Visualizar
Níveis	Draco Malfoy	1111131111	MH04 - Draco Malfoy	Editar Excluir Visualizar
	Fred Weasley	1111111112	-	Editar Excluir Visualizar
Razões de Desligamento	George Weasley	1111111113	-	Editar Excluir Visualizar
	Ginevra Weasley	1111111137	-	Editar Excluir Visualizar
Tipos de Matrícula	Harry Potter	1111111111	MH01 - Harry Potter	Editar Excluir Visualizar
	Hermione Granger	1111111115	MH02 - Hermione Granger	Editar Excluir Visualizar
	Luna Lovegood	1111111116	MH05 - Luna Lovegood	Editar Excluir Visualizar
	Neville Longbottom	1111111133	-	Editar Excluir Visualizar
	Romilda Vane	1111111136	-	Editar Excluir Visualizar
	Ronald Weasley	1111111117	MH03 - Ronald Weasley	Editar Excluir Visualizar
	Seamus Finnigan	1111111130	-	Editar Excluir Visualizar

16 Registros Encontrados 1 | Próximo

Figura 3.3: Tela da área de alunos

Antes de apresentar como está organizada a área de **Matrículas**, é importante explicitar a diferença entre aluno e matrícula no contexto do SAPOS. Como dito anteriormente, *aluno* é a entidade que guarda informações básicas de um aluno: nome, CPF, histórico de formação (cursos de especialização, graduação, mestrado e/ou doutorado realizados), entre outras. Já a *matrícula* é o identificador de um aluno em um determinado nível de curso (mestrado, doutorado, etc.). Assim, é importante observar a cardinalidade dessa associação: uma matrícula está associada a um e somente um aluno, mas um aluno possui uma ou mais matrículas. Isso acontece a fim de tornar possível o armazenamento do histórico acadêmico dos alunos formados no IC-UFF, ou seja, se um aluno fez mestrado e agora faz doutorado, sua matrícula referente ao mestrado continua armazenada e ele possui uma nova matrícula para o doutorado. Com isso, a entidade *matrícula* é o identificador de um aluno em um nível de formação. A área de matrícula, mostrada na Figura 3.4, é considerada a área central do sistema, já que nela é possível cadastrar todas as informações acadêmicas de um aluno. Devido ao seu grau de importância, a tela de Matrículas é a inicial do sistema, exibida logo após a autenticação do usuário no sistema.

A área de **Desligamentos** refere-se aos alunos que, por algum motivo, foram desligados do curso. Nesta tela, é possível associar um motivo de desligamento a uma matrícula explicitando a data do ocorrido. Alguns dos principais motivos de desligamento são: titulação, desistência e fim de prazo para a defesa.

SAPOS [ Sistema de Apoio à Pós-Graduação ] Instituto de Computação / UFF							Logout
Matrículas							Buscar Adicionar
Aluno	Número de Matrícula	Nível	Tipo de Matrícula	Data de Admissão	Desligamento		
Alunos >	Charlie Weasley	DH01	Doutorado	Regular	Junho-1995		Editar Excluir Visualizar
Desligamentos >	Bill Weasley	DH02	Doutorado	Especial	Março-1995		Editar Excluir Visualizar
	Harry Potter	MH01	Mestrado	Regular	Junho-1998		Editar Excluir Visualizar
Matrículas >	Hermione Granger	MH02	Mestrado	Regular	Março-1998		Editar Excluir Visualizar
	Ronald Weasley	MH03	Mestrado	Regular	Março-1998		Editar Excluir Visualizar
Níveis >	Draco Malfoy	MH04	Mestrado	Regular	Junho-1998		Editar Excluir Visualizar
Razões de Desligamento >	Luna Lovegood	MH05	Mestrado	Especial	Junho-1998		Editar Excluir Visualizar
7 Registros Encontrados							
Tipos de Matrícula >							

Figura 3.4: Tela da área de matrículas

Na seção de alunos foram adicionadas áreas onde são cadastradas informações que devem ser padronizadas, como: níveis, razões de desligamento e tipo de matrícula. Na área **Níveis** é possível cadastrar os níveis de formação (e.g. graduação, especialização, mestrado e doutorado) de um aluno da Pós-Graduação. A área **Razões de Desligamento** contempla todas as razões (e.g. desistência, prazo, titulação, entre outras) pelas quais um aluno pode ser desligado do curso de Pós-Graduação. A área **Tipo de Matrícula** foi criada para contemplar os diferentes tipos de ligação (e.g. especial e regular) entre um aluno e a Pós-Graduação. Isso permite que o sistema seja totalmente configurável, e seja facilmente adaptado a mudanças regimentais e de outra natureza, como, por exemplo, a criação de mais um curso. É importante ressaltar que, nessa tela, foram adicionadas apenas as áreas que são utilizadas pelos demais cadastros da seção alunos.

### 3.3 PROFESSORES

Esta seção armazena informações sobre dados básicos de professores, como nome e CPF. Além disso, relaciona o docente em questão aos alunos orientados ou coorientados por ele.

Na área **Professores**, como mostra a Figura 3.5, estão listados registros de professores, tais como nome e pontos de orientação. Cada aluno orientado corresponde a um ponto de orientação para o professor. Entretanto, quando o aluno possui um coorientador, somente meio ponto é computado tanto para o orientador principal como para o coorientador. Essa informação é importante, pois, segundo “Ata da Reunião do Colegiado da Pós-Graduação” (2011), um docente não pode ter mais que oito pontos de orientação. No entanto, essa

restrição ainda não é tratada no sistema, o que permite que um professor possua mais de oito pontos de orientação. Nesta área, ao clicar no botão visualizar de um registro, é possível verificar os alunos que estão sendo orientados por um determinado professor.

**SAPOS**  
[ Sistema de Apoio à Pós-Graduação ]  
Instituto de Computação / UFF

Logout

Versão 1.5.0

Alunos **Professores** Bolsas Etapas Formação Localidades Configurações

**Professores**  Buscar

	Nome	CPF	Data de Nascimento	Pontos de Orientação	
Professores >	Alastor Moody	22222222228	-	0,5	Editar Excluir Visualizar
Orientações >	Albus Dumbledore	22222222219	-	0,5	Editar Excluir Visualizar

**Filius Flitwick**  
 CPF: 22222222220  
 Data de Nascimento: -  
 Logradouro: -  
 Estado Civil: **Solteiro(a)**  
 Data de expedição da Identidade: -  
 Órgão Expeditor: -  
 Número da identidade: -  
 Bairro: -  
 Sexo: **Masculino**  
 Siape: -  
 1º Telefone: -  
 2º Telefone: -  
 CEP: -  
 Bolsas: **PH02**  
 Orientandos: **DH01 - Charlie Weasley, MH02 - Hermione Granger, MH05 - Luna Lovegood**

**Fechar**

Minerva McGonagall	22222222221	-	2,0	Editar Excluir Visualizar
Poppy Pomfrey	22222222230	-	0,0	Editar Excluir Visualizar
Remus Lupin	22222222222	-	0,0	Editar Excluir Visualizar
Severus Snape	22222222223	-	1,5	Editar Excluir Visualizar

7 Registros Encontrados

Figura 3.5: Tela da área de professores

Em **Orientações** (Figura 3.6), as matrículas dos alunos estão relacionadas com os docentes que os orientam. A informação referente a uma orientação estar ou não ativa também é exibida nesta área. Como dito anteriormente, existem dois tipos de relação de um professor com um aluno: orientador ou coorientador. A visualização de uma orientação exibe o orientador principal e uma lista de todos os professores que estão coorientando o aluno em questão.

**Orientações**

Professor	Número de Matrícula	Nome do Aluno	Orientador Principal	Orientação ativa?	Possui coorientador?	
Professor: <b>Albus Dumbledore</b> Matrícula: <b>MH01 - Harry Potter</b> Orientador Principal: <input checked="" type="checkbox"/> Lista de Coorientadores: <b>Alastor Moody , Minerva McGonagall , Severus Snape</b> <a href="#">Fechar</a>						
Filius Flitwick	MH02	Hermione Granger	<input type="checkbox"/>	Sim	Sim	<a href="#">Editar</a> <a href="#">Excluir</a> <a href="#">Visualizar</a>
Severus Snape	MH01	Harry Potter	<input type="checkbox"/>	Sim	Sim	<a href="#">Editar</a> <a href="#">Excluir</a> <a href="#">Visualizar</a>
Minerva McGonagall	MH01	Harry Potter	<input type="checkbox"/>	Sim	Sim	<a href="#">Editar</a> <a href="#">Excluir</a> <a href="#">Visualizar</a>
Minerva McGonagall	MH02	Hermione Granger	<input checked="" type="checkbox"/>	Sim	Sim	<a href="#">Editar</a> <a href="#">Excluir</a> <a href="#">Visualizar</a>
Alastor Moody	MH01	Harry Potter	<input type="checkbox"/>	Sim	Sim	<a href="#">Editar</a> <a href="#">Excluir</a> <a href="#">Visualizar</a>
Filius Flitwick	DH01	Charlie Weasley	<input checked="" type="checkbox"/>	Sim	Não	<a href="#">Editar</a> <a href="#">Excluir</a> <a href="#">Visualizar</a>
Severus Snape	MH04	Draco Malfoy	<input checked="" type="checkbox"/>	Sim	Não	<a href="#">Editar</a> <a href="#">Excluir</a> <a href="#">Visualizar</a>
Minerva McGonagall	DH02	Bill Weasley	<input checked="" type="checkbox"/>	Sim	Não	<a href="#">Editar</a> <a href="#">Excluir</a> <a href="#">Visualizar</a>
Filius Flitwick	MH05	Luna Lovegood	<input checked="" type="checkbox"/>	Sim	Não	<a href="#">Editar</a> <a href="#">Excluir</a> <a href="#">Visualizar</a>

10 Registros Encontrados

Figura 3.6: Tela da área de orientações

### 3.4 BOLSAS

Esta seção armazena informações sobre dados básicos de bolsas de fomento, como número de identificação da bolsa, agência de fomento, entre outras. Além disso, ela é responsável pela associação de uma bolsa de fomento a um aluno, através de sua matrícula.

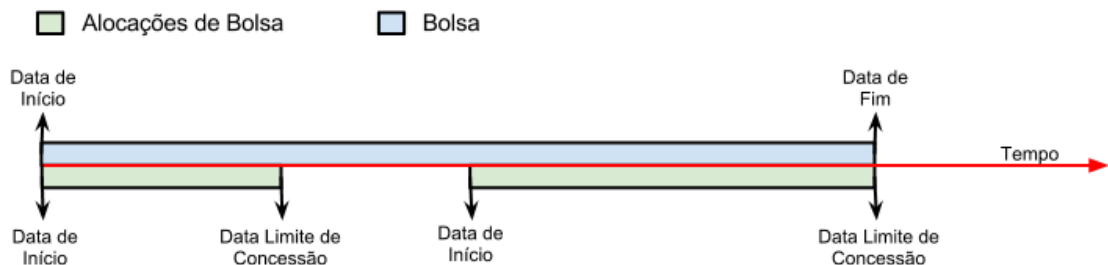
Uma bolsa de estudo possui diversas características. No SAPOS foram adicionadas, na área referente às **Bolsas**, apenas as características essenciais para o seu gerenciamento diário, o que inclui: número da bolsa, nível (e.g., mestrado ou doutorado), agência de fomento, tipo (e.g. Demanda Social, Nota 10 e REUNI) e as datas de início e fim. A data de início indica a data em que a bolsa foi concedida à instituição. A data de fim indica o prazo máximo em que a bolsa pode ser alocada. Normalmente, essa data não é preenchida, o que significa que a bolsa não tem um limite pré-estabelecido, e pode continuar sendo usada pela instituição sem prazo fixo. Além disso, é possível associar uma bolsa de estudos a um professor, caso a mesma tenha chegado à instituição por seu intermédio (e.g. bolsa de projeto).

Na **Alocação de Bolsas** é feita a associação entre a matrícula e a bolsa de estudo. Dessa forma, ao longo do ciclo de vida de uma bolsa, ela pode ser alocada a diversas matrículas. A alocação de uma bolsa a uma matrícula possui três datas: (i) a data de início refere-se ao período em que uma determinada bolsa foi associada a uma matrícula; (ii) a de

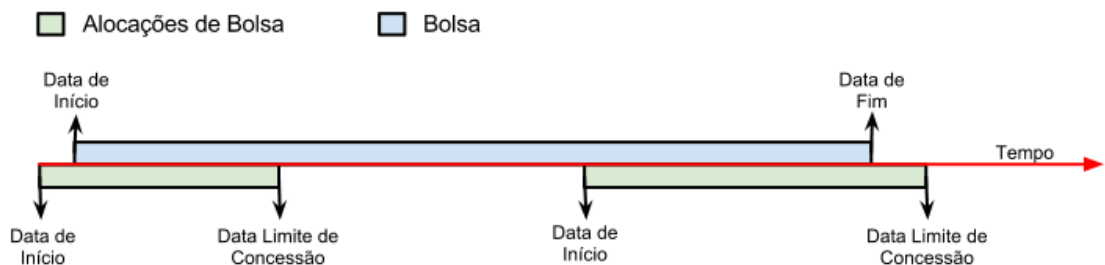
encerramento indica quando a associação de uma matrícula a uma determinada bolsa terminou de fato; e (iii) a de limite de concessão determina o prazo máximo que uma bolsa pode ficar associada a uma matrícula.

A matrícula do aluno pode ser alocada a uma bolsa por um determinado período, porém esta alocação segue as seguintes regras: (i) a bolsa não pode estar associada a outra matrícula no mesmo período  $p$  (ou mesmo em um período diferente que possua uma intersecção não vazia com o período  $p$ ); (ii) a data de início da bolsa deve ser anterior à data de início da alocação dessa bolsa; e (iii) a bolsa não pode estar expirada, ou seja, a data de fim da bolsa deve ser anterior à data de limite de concessão da alocação.

i)



ii)



iii)

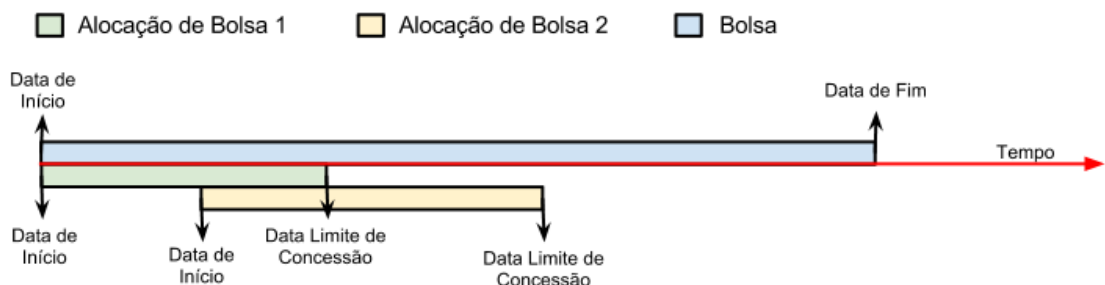


Figura 3.7: Validação de alocação de bolsas

As possíveis validações de alocação de bolsas estão representadas na Figura 3.7, que mostram as seguintes situações: (i) uma validação bem sucedida, onde todas as alocações de


uma bolsa estão obedecendo as regra de validação de data; (ii) uma validação mal sucedida, onde alocações de bolsa não obedecem os limites impostos; e (iii) uma validação mal sucedida, onde há a tentativa de se criar uma alocação de bolsa dentro do período de tempo de outra alocação de uma mesma bolsa.

Na seção Bolsas foram adicionadas as áreas onde são cadastradas informações que devem ser padronizadas, como Tipos de Bolsa e Agências de Fomento. Na área **Tipos de Bolsa** é possível cadastrar o programa (e.g. REUNI, Nota 10, Cota, entre outras) do qual a bolsa de fomento é proveniente. A área **Agências de Fomento** contempla o nome da agência que está fornecendo a bolsa de fomento. É importante ressaltar que foram adicionadas apenas as áreas que são utilizadas pelos demais cadastros da seção alunos.

### 3.5 ETAPAS

O curso de Pós-Graduação é composto por etapas obrigatórias. Esta seção tem o objetivo de prover as informações necessárias para que seja possível manter um controle das etapas realizadas por um aluno, através de sua matrícula.

O controle de etapas realizadas é importante para a área administrativa ter conhecimento da situação do aluno, já que essa circunstância está diretamente ligada à necessidade do aluno solicitar prorrogação. Na área de **Realização de Etapas**, como exibida na Figura 3.8, é possível estabelecer o cumprimento de uma etapa, associando a ela uma matrícula e uma data de conclusão.



The screenshot shows the SAPOS (Sistema de Apoio à Pós-Graduação) interface. The main content area is titled 'Realização de Etapas' and contains a form for 'Criar Realização de Etapas'. The form fields are: 'Etapa' (dropdown menu with 'Exame de Qualificação' selected), 'Matrícula' (text input), 'Data de conclusão' (dropdown for month 'Outubro' and year '2011'), and 'Observação' (text input). Below the form are 'Salvar' and 'Cancelar' buttons. A table below the form displays existing records:

Matrícula	Etapa	Data de conclusão	Observação	
MH04 - Draco Malfoy	Exame de Qualificação	Maio-2005	"Slytherin"	Editar Excluir Visualizar
MH02 - Hermione Granger	Exame de Qualificação	Maio-2012	"Gryffindor!"	Editar Excluir Visualizar
MH01 - Harry Potter	Exame de Qualificação	Maio-2005	"I'm not sure...Gryffindor!"	Editar Excluir Visualizar

At the bottom of the table, it indicates '3 Registros Encontrados'.

Figura 3.8: Tela de realização de etapas



A área de **Prorrogações** tem como objetivo controlar o tempo adicional que o aluno pode solicitar para concluir uma determinada etapa. Na tela de prorrogações, é possível associar uma matrícula a um tipo de prorrogação em uma determinada data. Atualmente, existem três tipos de prorrogação no PGC: regular, extraordinária e final (PGC, 2011), no entanto, na tela de **Tipos de Prorrogação** é possível editar ou definir novos tipos. Todo pedido de prorrogação deve ser solicitado ao colegiado e, quando aprovado, a data da aprovação é registrada no sistema.

### 3.6 FORMAÇÃO

A seção de **Formação**, exibida na Figura 3.9, contém informações básicas de instituições de ensino e os cursos que as compõem. O objetivo desta seção é evitar possíveis erros ou informações duplicadas, tornando, assim, consistentes os cadastros que as utilizam. Um exemplo de sua utilização pode ser visto no histórico de formação do discente, que utiliza os dados que foram cadastrados na área Formação.

O cadastro de instituições é composto pelo nome e a sigla da instituição. Já os cursos são cadastrados de acordo com o grau de formação e a instituição onde o mesmo é oferecido. A vinculação de um curso a uma instituição também pode ser feita no cadastro da instituição.

É possível associar um aluno a um curso de uma instituição, de acordo com seu histórico de formação. Vale lembrar que também é possível fazer essa associação através do cadastro de um novo aluno.

SAPOS [ Sistema de Apoio à Pós-Graduação ] Instituto de Computação / UFF Versão 1.5.2

Alunos Professores Bolsas Etapas **Formação** Localidades Configurações

**Instituições** Buscar Adicionar

Cursos > Instituições >

Nome Universidade Federal Fluminense  
Sigla UFF  
Cursos  
Ciência da Computação - Universidade Federal Fluminense - (Mestrado), Engenharia da Computação - Universidade Federal Fluminense - (Mestrado), Ciência da Computação - Universidade Federal Fluminense - (Doutorado)

Fechar

Universidade Federal do Rio de Janeiro	UFRJ	Editar Excluir Visualizar
Universidade de São Paulo	USP	Editar Excluir Visualizar
Universidade do Estado do Rio de Janeiro	UERJ	Editar Excluir Visualizar

4 Registros Encontrados

Figura 3.9: Tela da área de instituições e cursos

### 3.7 LOCALIDADES

Nesta seção estão listados os países, estados e cidades, como exibido na Figura 3.10, que são utilizados nas demais áreas do sistema. Da mesma forma que a seção Formação, o objetivo desta seção é de evitar possíveis erros ou informações duplicadas, tornando, assim, consistentes os cadastros que as utilizam.

Em um primeiro momento, esta seção seria composta apenas pelo cadastro de **estados** e **cidades**. Entretanto, devido à existência de alunos estrangeiros, optou-se por adicionar também o **país** de origem. Com isso, foi possível definir a associação entre país, estados e cidades.

Nome	Sigla	País
Rio de Janeiro	RJ	Brasil
São Paulo	SP	Brasil

2 Registros Encontrados

Figura 3.10: Tela de localidades

### 3.8 CONFIGURAÇÕES

Esta seção tem o objetivo de controlar o acesso ao sistema a partir de configurações definidas pelos administradores. Atualmente, a seção comporta a administração de usuários, como exibido na Figura 3.11. Nessa área é possível realizar a atualização de nomes e senhas, o cadastro de novos usuários e a exclusão dos mesmos.

Nome do usuário
admin

1 Registro Encontrado

Figura 3.11: Tela de configurações

É interessante observar que, dado o número reduzido de usuários do SAPOS, esta seção contempla o mínimo necessário para o controle de acesso. No entanto, com o aumento do número de usuários, esta seção deve evoluir para um controle de perfis e permissões de acesso, possibilitando uma maior personalização da característica administrativa do SAPOS.

### **3.9 CONSIDERAÇÕES FINAIS**

Neste capítulo foi apresentada a forma como o SAPOS está organizado, com uma descrição de cada tela e suas funcionalidades. Em cada seção foi discutida a motivação para sua criação e suas principais áreas. Além disso, foram apresentados exemplos de telas do sistema, a fim de, exemplificar suas funcionalidades. O próximo capítulo discute a metodologia de desenvolvimento utilizada.

## CAPÍTULO 4 – DESENVOLVIMENTO

Antes de iniciar o desenvolvimento de um sistema, é necessário definir a plataforma sobre a qual este será implementado. A utilização da plataforma *web* para o desenvolvimento do SAPOS deve-se às diversas vantagens que aplicações *web* possuem para os usuários e desenvolvedores, por exemplo, a evolução da aplicação ser transparente ao usuário, possibilidade de integração com outros sistemas, maior interatividade com o usuário final, disponibilidade de acesso a partir de diferentes locais e independência de plataforma.

Além disso, é necessário realizar outras escolhas, por exemplo, a linguagem de programação utilizada e o método de desenvolvimento que será aplicado. No SAPOS, essas decisões também foram consideradas, e acabou-se optando por utilizar uma metodologia ágil de desenvolvimento e a linguagem de programação Ruby.

Outro ponto que deve ser levantado é a migração de dados. Os sistemas legados eram baseados em arquivos *Microsoft Access* e planilhas *Microsoft Excel* nomeadas por data. Essa organização dificultava a consulta de dados, criava versões diferentes de arquivos e era propensa a duplicação de informação. Devido a esses problemas, tornou-se necessária a centralização dos dados.

Este capítulo está organizado como se segue. A Seção 4.1 apresenta os conceitos da metodologia ágil e sua aplicação no SAPOS. A Seção 4.2 descreve os padrões de projeto utilizados neste trabalho e justifica a escolha deles. A Seção 4.3 apresenta a linguagem de programação utilizada e suas vantagens. Na Seção 4.4 é explicitado o processo de migração de dados. A Seção 4.5 discute a implementação e aborda os principais marcos do desenvolvimento do sistema. Por fim, a Seção 4.6 apresenta considerações finais sobre o capítulo.

### 4.1 METODOLOGIA ÁGIL

De acordo com PRESSMAN (2004), a filosofia pregada para o desenvolvimento de sistemas é a de que o processo de desenvolvimento deva ser perfeitamente previsível, podendo ser planejado, estimado e completado com sucesso. Entretanto, na prática, o ambiente de desenvolvimento costuma ser confuso e sofre constantes mudanças, o que causa grande impacto no produto final.

O processo de desenvolvimento ágil propõe uma mudança de paradigma, onde o foco está na entrega constante de software utilizável para o usuário (COHEN; LINDVALL; COSTA, 2004). A fim de alcançar esse objetivo, segundo COHEN (2004), podem-se destacar

como principais características: (i) uma maior colaboração entre programadores e especialistas do domínio; (ii) comunicação constante com o cliente; (iii) equipes auto gerenciáveis; e (iv) adaptação do código e da equipe às mudanças dos requisitos em qualquer etapa do desenvolvimento.

Tendo em vista a preferência por comunicação constante com o cliente, os métodos ágeis normalmente produzem menos documentação quando comparados a outros métodos. É notável também a grande capacidade de adaptação dos projetos em relação às mudanças, o que em grande parte dos métodos tradicionais não acontece.

#### 4.1.1 APLICAÇÃO DE MÉTODOS ÁGEIS NO SAPOS

De acordo com o que foi apresentado, existem benefícios na utilização de um método de desenvolvimento ágil. Neste trabalho foram utilizadas diversas práticas ágeis, como: definição de um *sprint*, reuniões de planejamento de um *sprint*, reunião para apresentação do que foi produzido no *sprint* e reuniões para avaliação do processo ao término de cada *sprint*. Nesta seção, serão explicitadas as práticas utilizadas no desenvolvimento do SAPOS e os artefatos utilizados para apoiar as mesmas.

Os *sprints*, segundo SCHWABER (2004), são ciclos de desenvolvimento de duração fixa, que têm como principal característica a entrega de um produto incrementado e pronto para a utilização. Entretanto, a teoria sugere apenas o tamanho máximo para um *sprint*, pois o tamanho de um *sprint* deve ser definido pelo tempo necessário para que seja possível desenvolver algo tangível para o cliente. Para o desenvolvimento do SAPOS, optou-se por começar com um tamanho de *sprint* de duas semanas e, após o início do desenvolvimento, constatou-se que esse período foi adequado para entregar uma versão operacional do produto ao final de cada *sprint*.

Antes de descrever as reuniões de planejamento, é necessário apresentar um artefato importante do desenvolvimento ágil, o *backlog* do produto. “O *backlog* do produto é uma lista ordenada de tudo o que pode ser necessário no produto e é a fonte única dos requisitos para qualquer mudança a ser feita no produto” (SCHWABER, 2004). A teoria não define um método de ordenação padrão, mas elucida a necessidade da existência de um método que priorize a demanda de maior importância para o cliente. No desenvolvimento do SAPOS, foi utilizado um *backlog* do produto com regras para a priorização das demandas que eram levantadas junto aos envolvidos no projeto.

A reunião de planejamento do *sprint*, de acordo com SCHWABER (2004), tem como objetivo responder duas perguntas: o que vai ser entregue no próximo *sprint* e como será

realizado o trabalho necessário para entregá-lo. O resultado obtido é um planejamento simples e de curto prazo. No SAPOS, a reunião de planejamento foi dividida em duas partes, cada uma com cerca de uma hora de duração. Na primeira parte, o *backlog* do produto era atualizado com as novas demandas e reorganizado, priorizando as necessidades mais importantes para o usuário. A seguir eram escolhidas pelo usuário as demandas que iriam ser desenvolvidas durante o *sprint*, compondo o *backlog* de *sprint*. Na segunda parte, as demandas incluídas no *backlog* de *sprint* eram divididas em atividades necessárias para o seu desenvolvimento.

Nos métodos ágeis, existe uma reunião chamada revisão do *sprint*, onde o cliente avalia o que foi desenvolvido com base no que foi descrito na especificação de requisitos do produto. Entretanto, foi constatado que essa reunião traria muita burocracia ao processo de desenvolvimento do SAPOS. Optou-se então por uma reunião mais simples, onde a cada novo lançamento de versão eram apresentadas as novas funcionalidades do produto. Além disso, essa reunião era utilizada para levantar novos requisitos para inserção no *backlog* do produto.

Por fim, o último conceito a ser apresentado é o de retrospectiva *do sprint*. Esse é um momento reservado pela equipe de desenvolvimento para inspecionar o processo, apontando falhas e propondo melhorias para os próximos *sprints*, como explicitado por SCHWABER (2004). No SAPOS, a *retrospectiva* era realizada a cada *sprint* e tinha a duração máxima de trinta minutos. Nessa reunião foram definidos pontos importantes de melhoria, como: (i) mudança do horário da reunião de planejamento; (ii) definição de uma ferramenta para a comunicação on-line; e (iii) programação em conjunto em situações de dificuldade.

## 4.2 PADRÃO DE PROJETO

O padrão MVC, ou Modelo - Visão - Controle, é um estilo arquitetural composto por três camadas, como exibido na Figura 4.1. Nesta notação, as caixas representam a separação de funções das três camadas e as setas exibem como é realizada a comunicação entre elas, cada uma exercendo um papel dentro do sistema (KRASNER; POPE, 1988).

A utilização de um padrão arquitetural permite aplicar diretamente a separação de funcionalidades, ou seja, evitar a sobreposição de funções de cada módulo do sistema. A aplicação do MVC é diretamente associada ao *framework* Ruby on Rails, que será abordado na Seção 4.3.2.

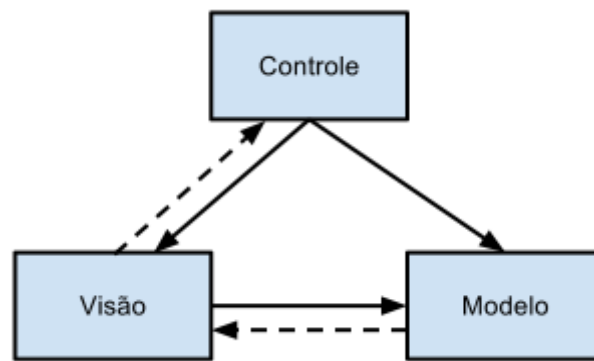


Figura 4.1: Uma visão abstrata da arquitetura MVC

#### 4.2.1 MODELO

A camada de modelo representa as entidades do domínio, que contêm informações usualmente persistidas em um banco de dados, bem como as regras de negócio referentes a elas. Além disso, ela também é vista como a camada que possui a lógica da aplicação, pois fornece a base de regras de cada tipo de objeto declarado em um sistema, ou seja, o modelo é o principal responsável por fornecer métodos à aplicação.

Todo dado acessado pela aplicação, inclusive seu armazenamento em um banco de dados, é de responsabilidade da camada de modelo. Como observado no SAPOS, todos os modelos que são persistidos no sistema, como a entidade *Professores*, são de responsabilidade da camada em questão.

#### 4.2.2 VISÃO

A camada de visão, também conhecida como camada de apresentação, desempenha o papel de interface da aplicação. Essa camada não possui conteúdo relacionado à lógica de negócios, pois o processamento é feito pelas camadas de controle e modelo, que por sua vez, redirecionam para a visão.

Outra característica da camada de visão é criar a interface de entrada e saída de dados gerados pela camada de controle. A visão deve garantir que a apresentação dos dados reflita o estado do modelo, ou seja, quando os dados do modelo são atualizados, é necessário notificar as classes da camada de visão que dependem do mesmo. No SAPOS, todas as telas exibidas ao usuário são de responsabilidade da camada de visão.

#### 4.2.3 CONTROLE

A camada de controle é responsável por processar, interpretar e responder requisições vindas das outras duas camadas, coordenando a comunicação entre elas. No SAPOS, a

camada de controle define todo processamento de informações que são inseridas na camada de visão e a conversão para objetos da camada de modelo para armazenamento no banco de dados.

#### **4.2.4 INTERAÇÃO ENTRE AS CAMADAS**

A separação em camadas do MVC proporciona diversas vantagens, por exemplo, separar as regras de negócio da camada de visão, evitar repetição de código, manter uma boa coesão e aumentar a possibilidade de reuso de código. Cada uma das três camadas do MVC representa uma separação conceitual entre componentes do sistema, facilitando a modularidade e o isolamento funcional.

A separação conceitual do padrão traz a necessidade da comunicação entre camadas, como representado pelas setas sólidas e pontilhadas na Figura 4.1. As setas sólidas representam a comunicação entre duas camadas, enquanto as setas pontilhadas refletem a resposta a mensagens enviadas pelas camadas.

Alguns exemplos de mensagens trocadas entre as camadas são: (i) comandos enviados pelo controlador para uma visão ou um modelo associado a ele; (ii) notificação, pelo modelo, de visões associadas quando ocorre mudança de estado; (iii) e requisições da camada de visão para a camada de modelo para que sejam representadas informações do modelo na camada de visão.

### **4.3 LINGUAGEM DE PROGRAMAÇÃO**

A definição de uma tecnologia em um projeto varia conforme o objetivo do software a ser desenvolvido. É importante definir uma linguagem de programação com características que solucionem os problemas impostos pela definição do sistema e que possam facilitar seu desenvolvimento. Linguagens confiáveis, estáveis, que possuam documentação e uma comunidade ativa, podem ser consideradas boas escolhas. Baseando-se nesses fatores, decidiu-se por utilizar o Ruby como linguagem de programação neste projeto.

#### **4.3.1 RUBY**

A linguagem de programação Ruby é uma linguagem orientada a objetos, interpretada e de propósito geral. A linguagem criada por Yukihiro Matsumoto tem como objetivo ser interativa ao desenvolvedor, possuindo as características de fácil leitura e escrita. O Ruby é uma linguagem dinâmica que combina a sintaxe das linguagens *Perl*, *Smalltalk*, *Eiffel* e *Lisp* (MATSUMOTO, [S.d.]).



A linguagem considera que todos seus elementos são objetos, ou seja, todos os elementos possuem atributos e métodos. Tipos primitivos são objetos instanciados automaticamente pela linguagem e possuem seus próprios métodos, como exibido na Figura 4.2.

```
5.times { puts "Ruby Language" }
```

Figura 4.2: Trecho de código que imprime cinco vezes o texto “*Ruby Language*”

Uma característica importante da linguagem é a fácil importação de bibliotecas de terceiros, que são chamadas de *gems* (QUARANTO, [S.d.]). As *gems* são armazenadas em um repositório disponibilizado após a instalação do interpretador da linguagem e qualquer usuário pode colaborar com o desenvolvimento e extensão do repositório. No SAPOS, são utilizados alguns desses *gems*, posteriormente descritos na Seção 4.3.4.

### 4.3.2 RUBY ON RAILS

O *Ruby on Rails* é um framework *web*, implementado em *Ruby*, que inclui um conjunto de ferramentas necessárias para a construção de aplicações *web* seguindo o padrão MVC sob uma base de dados. O *Ruby on Rails* baseia-se em um conjunto de conceitos, dentre eles:

- COC (*Convention Over Configuration*, ou Convenção Sobre Configuração): visa estabelecer um padrão de nomenclatura, regras e organização a fim de diminuir o tempo gasto com a configuração da aplicação.
- DRY (*Don't Repeat Yourself*, ou Não Se Repita): o princípio DRY é baseado no reuso e na boa coesão de escrita. A modificação de um trecho de código não deve implicar na alteração de outros que são logicamente independentes, permitindo assim um alto reaproveitamento de funcionalidades.
- RESTFul (*Representational State Transfer*, ou Transferência do Estado Representacional): o termo apresentado por FIELDING (2000), é baseado em dois princípios: o uso de identificadores de recurso (*Uniform Resource Locator* ou URL) e a mudança de estados de um objeto utilizando métodos do protocolo HTTP, como os métodos GET, POST, PUT e DELETE.
- ORM (*Object-Relational Mapping*, ou Mapeamento Objeto-Relacional): técnica de desenvolvimento que abstrai comandos SQL e associa métodos a objetos da camada de modelo, criando objetos de domínio persistentes.

A escolha do Ruby on Rails para a implementação do SAPOS considerou o poder de adaptação do framework, sendo ideal para o alinhamento com as técnicas ágeis escolhidas para o projeto. Esta característica do framework, de adicionar facilmente novas funcionalidades, resultou na utilização de um *sprint* de duas semanas, como mencionado anteriormente, devido à velocidade de implementação.

Sob a definição da arquitetura MVC, o *Ruby on Rails* distribui papéis a cada uma de suas camadas. Cada papel possui um conjunto de atribuições. Por exemplo, a camada de visualização tem como principal atribuição processar os arquivos de *template* (arquivos com a extensão **.erb**), gerando um arquivo HTML capaz de ser interpretado pelo navegador. Por outro lado, a camada de controle é responsável por processar requisições e fornecer respostas, produzindo diversos tipos de dados (apresentados na Figura 4.3) solicitados pelo usuário. Para apoiar na interação entre as camadas de controle e visualização, o *Ruby on Rails* possui um conjunto de classes nomeado de *ActionPack* (HANSSON, [S.d.]), que é responsável por preparar uma resposta para uma requisição feita ao servidor.

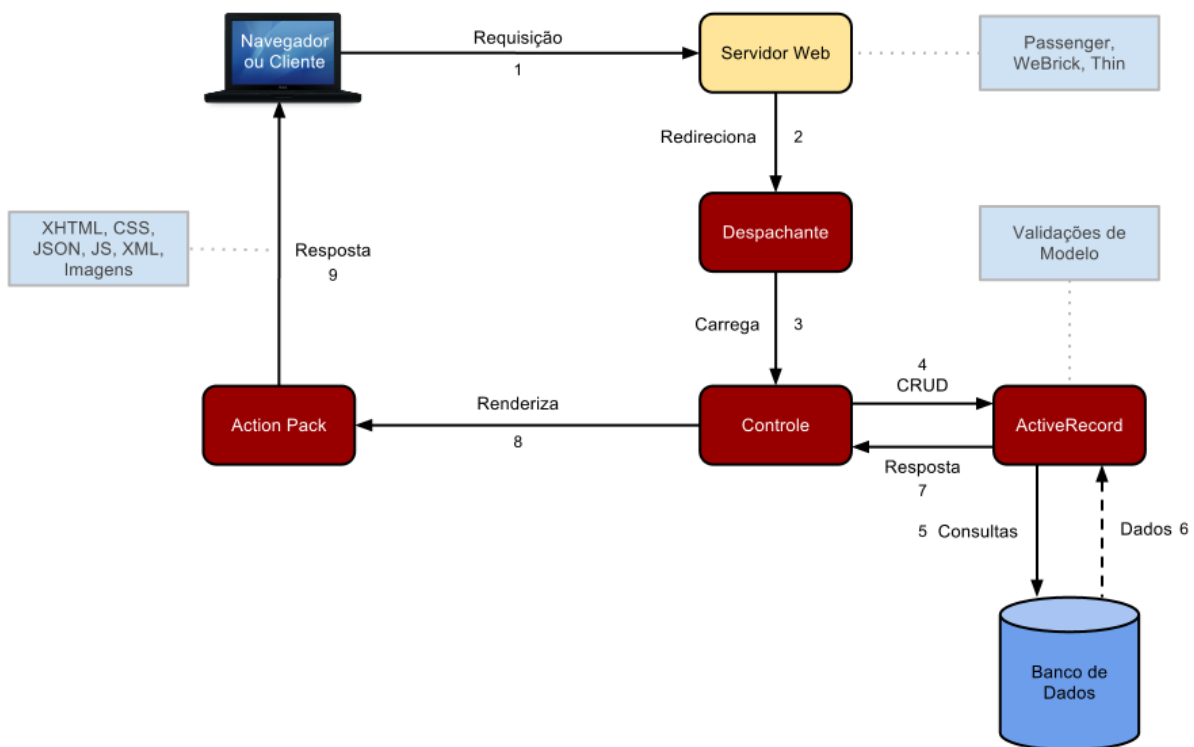


Figura 4.3: Arquitetura do Ruby on Rails

Representando a camada de modelo, temos o componente *ActiveRecord* (HANSSON, [S.d.]), que tem como principal funcionalidade conectar objetos e tabelas de bancos de dados

para criar um modelo de domínio persistente. A lógica e os dados são representados unicamente, sendo implementados como um padrão de mapeamento objeto-relacional (ORM).

A modelagem da arquitetura MVC, bem como os componentes do *Ruby on Rails*, podem ser observados na Figura 4.3, que mostra o fluxo feito por uma requisição sobre uma aplicação. Como é possível observar na Figura 4.3, o ciclo se inicia quando o usuário acessa uma página do sistema. Nesse momento a camada de controle é acionada e a requisição então é processada pelo servidor HTTP que hospeda a aplicação, para que seja possível interpretar o que foi pedido pelo cliente. Caso existam pedidos de consulta ou escrita à camada de modelo, os pedidos são redirecionados para o componente *ActiveRecord*, para que seja realizada a consulta ao banco de dados. Após o controle receber a resposta da camada de modelo, ele o redireciona para o *ActionPack*, que é responsável pelo formato de resposta da camada de visão, possibilitando o envio de vários formatos como HTML, CSS, JavaScript, JSON, XML, entre outros.

### 4.3.3 PRINCIPAIS BIBLIOTECAS UTILIZADAS

A utilização de bibliotecas externas, ou *gems*, foi necessária devido ao surgimento de diferentes requisitos ao longo do desenvolvimento do sistema, tais como gerenciamento e busca de dados, validações de data e geração de relatórios em PDF.

O *ActiveScaffold* (RUTHERFORD, [S.d.]) é uma *gem* do *Ruby* que lida com a representação de modelos de forma dinâmica e interativa, facilitando a configuração de elementos da camada de visualização e controle. Utilizando a tecnologia AJAX (*Asynchronous Javascript and XML* ou XML e *Javascript* Assíncronos), o *ActiveScaffold* gera uma interface tabular para lidar com os dados e todo o seu CRUD (*Create, Read, Update and Destroy* ou Criação, Leitura, Atualização e Destruição), adicionando à visualização elementos de ordenação, paginação e busca. A geração do CRUD realizado pelo *ActiveScaffold* é feita a partir de comandos, como exibidos na Figura 4.4, onde é realizada a concepção da interface da entidade **Instituição**.

```
rails generate active_scaffold Institution name:string
```

Figura 4.4: Comando de geração utilizado pelo *ActiveScaffold*

Essa *gem* é configurável em sua interface (HTML, *JavaScript* e CSS) e compatível com a maioria dos navegadores *web*. Vale ressaltar que a *gem* em questão é um projeto de código aberto. A utilização de uma *gem* com as características do *ActiveScaffold* foi essencial,

pois foi o principal suporte tanto da interface quanto da navegação. Seu poder de configuração possibilitou modificações necessárias ao longo do desenvolvimento deste projeto.

Seguindo o princípio DRY, explicitado na Seção 4.3.2, foram utilizadas *gems* que solucionam problemas de validação de datas e geração de relatórios em PDF. Para a geração de relatórios foi utilizada a *gem Prawn* (GREGORY; HEALY, [S.d.]), que é uma biblioteca de edição de PDF, com o intuito de auxiliar na construção de imagens, tabelas e textos personalizados e exportação do documento.

Outra *gem* empregada foi a *validates\_timeliness* (MEEHAN, [S.d.]), que é utilizado para validação de datas diretamente no modelo, como exibido na Figura 4.5. Essa *gem* pode utilizar mensagens em diversas línguas, diferentes fusos horários e conversão automática de formatos de data. A *gem* foi utilizada na solução do problema de validação de alocação de bolsas a uma matrícula, devido as situações de conflito de datas de criação e fim detalhadas na Seção 3.4 do Capítulo 3.

```
validates_date :start_date, :on_or_after => :scholarship_start_date
```

Figura 4.5: Trecho de código para validação de datas em alocação de bolsas

## 4.4 MIGRAÇÃO DE DADOS

Como dito anteriormente, a Pós-Graduação em Computação já fazia uso de arquivos *Microsoft Access* e planilhas *Microsoft Excel* quando se iniciou o desenvolvimento do SAPOS. Devido ao SAPOS utilizar um banco de dados próprio, a importação de dados anteriormente cadastrados foi importante. A migração para uma estrutura centralizada visava evitar a duplicação de registros e a criação de arquivos desnecessários, possuindo um maior controle sobre os dados.

### 4.4.1 FERRAMENTA UTILIZADA

A partir da versão 2.3.4, apresentada em HANSSON (2009), o *Ruby on Rails* disponibilizou uma ferramenta para a inserção rápida de dados em um banco de dados utilizando suas próprias classes e camada de modelo chamada *seed*. A técnica tem como objetivo disponibilizar uma primeira carga de dados a um sistema recém-instalado. Essa técnica tem como base um arquivo denominado *seeds.rb*, que contém as informações necessárias para uma primeira carga no banco de dados.

Apesar da ferramenta *seed* geralmente ser utilizada para uma primeira carga de dados na aplicação, ela também pode ser utilizada para a população de banco de dados para um

cenário de migração, onde muitos dados precisam ser importados de forma que não se perca a consistência da informação. O conteúdo do arquivo *seed* possui a mesma sintaxe do *Ruby* e referencia diretamente todas as classes do *ActiveRecord*, acessando métodos que têm acesso direto ao banco de dados, além de possibilitar a criação de funções que podem auxiliar no processo de migração.

Devido ao grande volume de dados presente nos bancos de dados anteriores à implantação do SAPOS, foram utilizadas planilhas com partes fixas escritas em *Ruby*, a fim de facilitar a montagem do *seeds.rb*. Assim, foi necessário escrever consultas SQL nos arquivos *Microsoft Access* para extrair os dados que deveriam ser importados. Em posse da tabela resultante da consulta executada no *Microsoft Access*, foram copiadas as colunas contendo os dados que deveriam ser inseridos no novo banco de dados. A seguir, esses dados foram adicionados à planilha, como pode ser visto na Figura 4.6, permitindo a criação do arquivo executável *seeds.rb*.

Country.create(	:name =>	" Líbia	" )
Country.create(	:name =>	" Liechtenstein	" )
Country.create(	:name =>	" Lituânia, República da	" )
Country.create(	:name =>	" Luxemburgo	" )
Country.create(	:name =>	" Macau	" )
Country.create(	:name =>	" Macedônia	" )
Country.create(	:name =>	" Madagascar	" )
Country.create(	:name =>	" Madeira, Ilha da	" )
Country.create(	:name =>	" Malásia	" )

Figura 4.6: Parte da tabela de países para migração de dados

Por fim, foram geradas as tabelas de matrículas (485 tuplas), alunos (485 tuplas), Professores (37 tuplas), orientações (241 tuplas), instituições (85 tuplas), cursos (159 tuplas), países (245 tuplas), estados (31 tuplas) e cidades (9.713 tuplas). Ao final desse processo, foi gerado um total de 11.481 tuplas distribuídas entre 9 entidades.

## 4.5 IMPLEMENTAÇÃO DO SISTEMA

O processo de desenvolvimento do SAPOS fez uso de características ágeis, como descrito na Seção 4.1. Nesse processo, existiram as etapas de levantamento de requisitos, modelagem, desenvolvimento e testes para cada *sprint*. Essas características proporcionaram a evolução do sistema, de acordo com o andamento dos *sprints*.

### 4.5.1 FERRAMENTAS UTILIZADAS

Após a introdução do sistema na Pós-Graduação, foi necessário criar um canal de comunicação entre usuários e desenvolvedores, para que fosse possível centralizar a troca de

mensagens, evitando a perda de informação. Dada essa necessidade, foi adotado o *Redmine*<sup>1</sup>, um software de gerência de projetos que possui diversas ferramentas, como: rastreamento de defeitos, requisições de novas funcionalidades, calendários, fóruns de discussão e notificações de atualizações.

Para apoiar a construção do SAPOS, foi utilizado o *Git*<sup>2</sup>, um sistema de controle de versão e gerenciamento de código que possui a ideia de que toda máquina que contém um projeto pode ser tratada como um repositório independente. A utilização deste gerenciador de código viabilizou o desenvolvimento em um ambiente controlado, possibilitando a participação de vários integrantes em um só projeto e o uso de ferramentas como comparação de código, versionamento e controle de colisões.

O projeto contou com o uso de um repositório central inicialmente hospedado no Laboratório de Engenharia de Software (SEL) utilizando o software *Gitorious*<sup>3</sup>, que serviu de ponto central de hospedagem de projetos. Ao longo do desenvolvimento do projeto, a sua hospedagem foi migrada para o *GitHub*<sup>4</sup>, diminuindo assim a carga do servidor localizado no SEL e transferindo o projeto para a nuvem.

O SAPOS, por se tratar de um sistema web, necessitou ser hospedado em um servidor (localizado no SEL) para o acesso de seus usuários. A passagem do ambiente de desenvolvimento para o de produção necessitava de um acesso remoto aos servidores da UFF e ao servidor onde o SAPOS estava hospedado, realizados através de protocolos *Secure Shell* (YLONEN; LONVICK, [S.d.]).

A atualização do sistema em ambiente de produção, como mencionado na introdução do Capítulo 4, era transparente ao usuário devido ao servidor HTTP (*Phusion Passenger*<sup>5</sup>) receber os arquivos da nova versão do sistema e apenas reiniciar a aplicação Ruby on Rails quando requisitado (PHUSION, [S.d.]). O processo era realizado automaticamente através da execução de um roteiro escrito em *Shell Script* (KERNIGHAN; PIKE, 1984), que executava uma busca no *Git* da versão descrita em sua chamada e então efetuava uma cópia do estado dos arquivos com o mesmo rótulo do *Git*. Tal versão era utilizada para exibir a versão corrente do sistema para os usuários. Por exemplo, se quiséssemos atualizar o sistema para uma versão marcada com o rótulo 1.0, executaríamos o *script* com a seguinte chamada: “*./update-sapos.sh 1.0*”.

---

<sup>1</sup> (LANG, [S.d.])

<sup>2</sup> (TORVALDS, [S.d.])

<sup>3</sup> (SORENSEN, [S.d.])

<sup>4</sup> (WANSTRATH; HYETT; PRESTON-WERNER, [S.d.])

<sup>5</sup> (PHUSION, [S.d.])

Para a implementação, inicialmente foi utilizado o ambiente integrado de desenvolvimento *NetBeans*<sup>6</sup>, posteriormente substituído pelo *RubyMine*<sup>7</sup>. O *NetBeans* é uma IDE para desenvolvimento em diversas linguagens de programação, incluindo Ruby. Porém, o suporte da IDE à linguagem foi abandonado na versão 7.0 (“Net Beans Community News”, [S.d.]), motivando assim a migração para um novo ambiente.

A escolha do *RubyMine* levou em consideração a sua integração ao ambiente *Ruby on Rails*. O software oferece um conjunto de ferramentas para a conexão com o framework *web*, possibilitando acionar o terminal de comandos do framework, consultar e atualizar bibliotecas externas (*gems*), consultar documentação de métodos e integração com sistemas de controle de versões (principalmente *git*).

#### 4.5.2 HISTÓRICO DE VERSÕES

O SAPOS, em sua primeira versão, contemplava o relacionamento entre aluno, matrícula, professor e bolsas (o núcleo do sistema), como pode ser visto na Figura 4.7. O objetivo dessa primeira versão foi resolver o principal problema reportado pela Pós-Graduação: o controle de bolsas de fomento. Entretanto, essa versão ainda não estava apta para a utilização dos usuários, pois carecia dos campos necessários e dos dados armazenados nos sistemas legados.

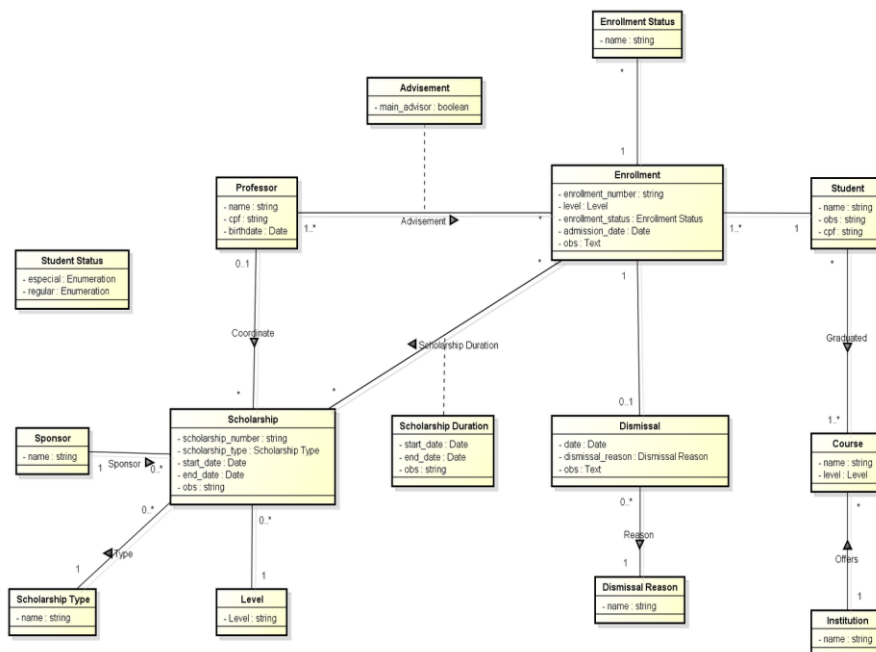


Figura 4.7: Primeira versão do modelo do sistema

<sup>6</sup> (ORACLE, [S.d.])

<sup>7</sup> (JETBRAINS, [S.d.])

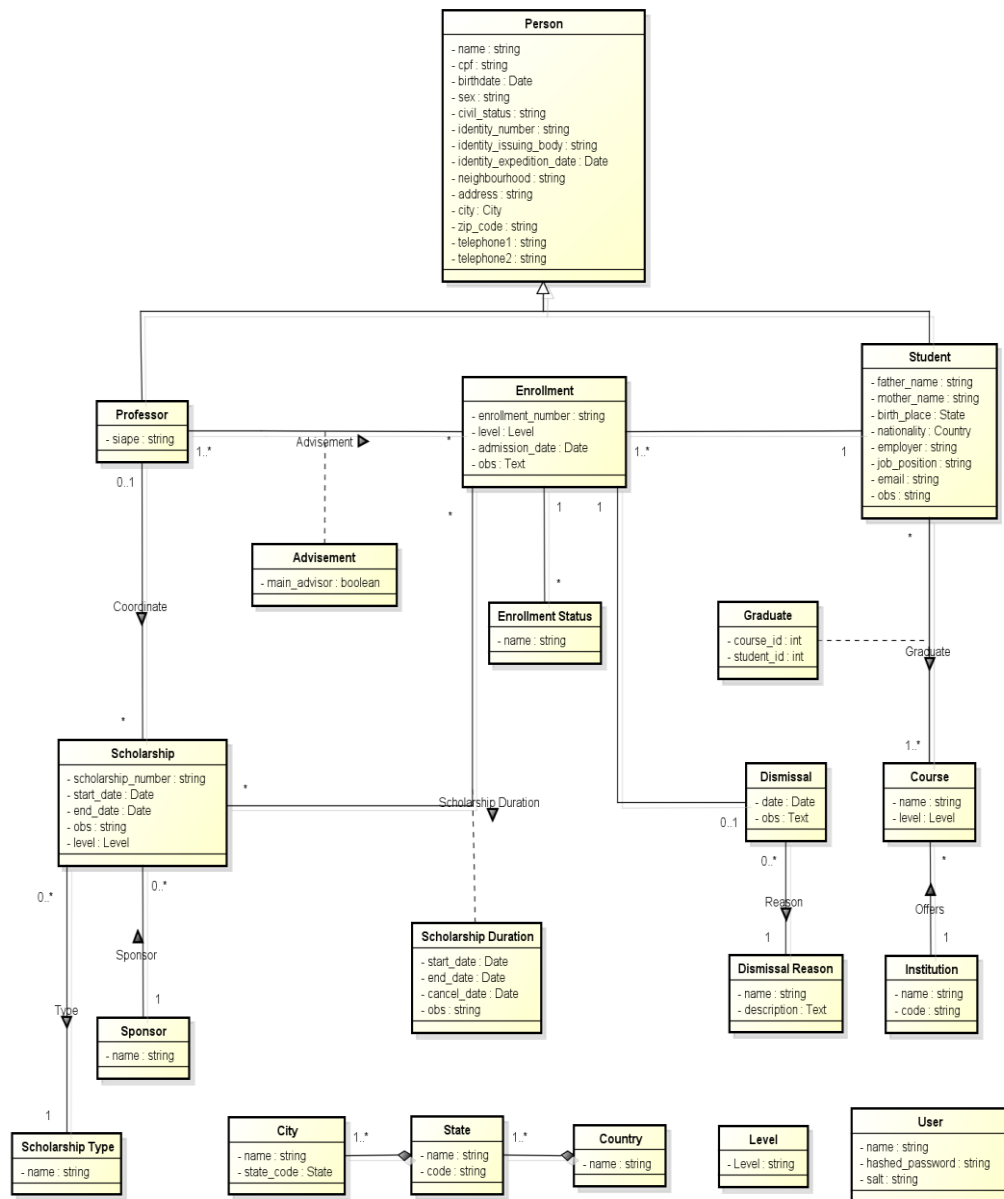


Figura 4.8: Modelo final completo com todos os atributos

Na segunda versão, foram adicionadas ao sistema as informações necessárias para que o mesmo pudesse ser colocado em produção, como pode ser visto na Figura 4.8. Assim, após uma análise dos bancos de dados utilizados pelos sistemas anteriores, foram definidos os campos que deveriam constar no SAPOS. Além disso, para que não houvesse perda de informação na implantação do novo sistema, foi realizado o processo de migração de dados. Esse processo foi responsável por centralizar no SAPOS as informações dos vários sistemas utilizados na Pós-Graduação, como visto na Seção 4.4. Vale ressaltar que, no início, o SAPOS



foi utilizado em paralelo com os demais sistemas e, conforme as funcionalidades eram adicionadas, ele passava a se tornar o sistema principal para tratar das funcionalidades em questão.

As versões posteriores foram divididas em dois grupos: correção e evolução. As versões de correção mostraram quantas vezes foi necessário lançar uma versão emergencial corrigindo um erro que não poderia esperar até o final do *sprint*. As versões de evolução contemplaram as mudanças que tinham como resultado a adição de novas funcionalidades.

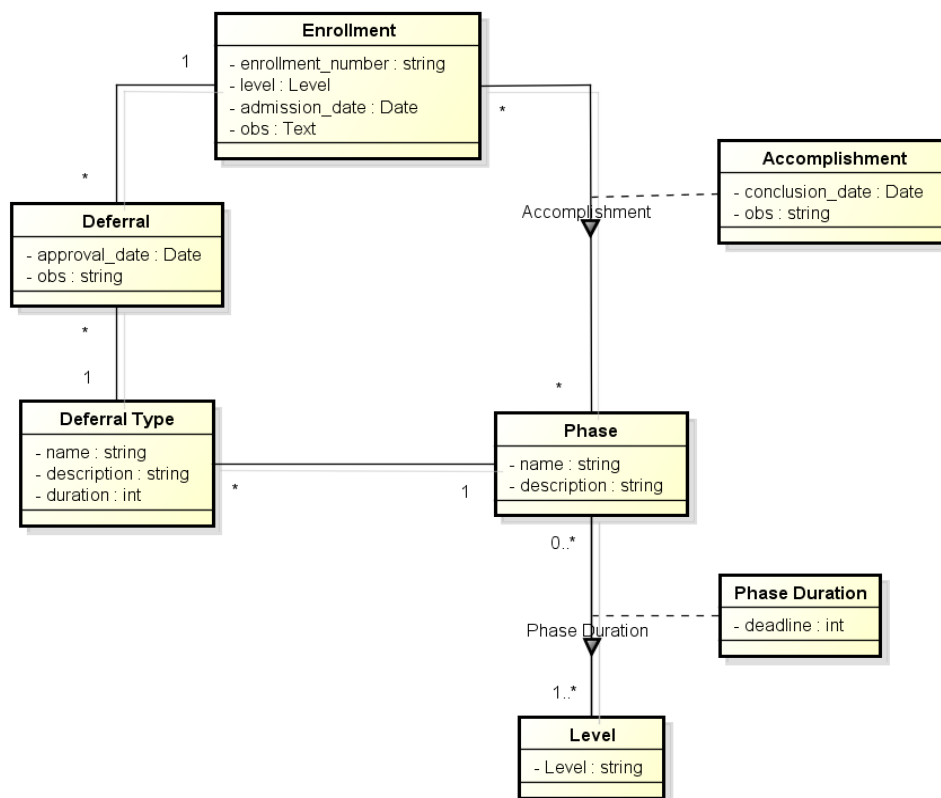


Figura 4.9: Modelo de etapas e controle de prorrogações

Antes de apresentar as versões posteriores é necessário explicitar o método utilizado para rotular as versões do sistema. O rótulo de versão do SAPOS é composto por três números separados por pontos. O critério utilizado para rotular as versões foi incrementar a unidade do algarismo menos significativo para representar o lançamento de uma versão corretiva, e utilizar o segundo algarismo menos significativo para representar o lançamento de versões evolutivas. O algarismo mais significativo foi utilizado para representar grandes mudanças no sistema, por exemplo, sua entrada no ambiente de produção. Vale ressaltar que ao incrementar o algarismo mais significativo, os algarismos subsequentes eram zerados.

A primeira versão lançada após a implantação do SAPOS na Pós-Graduação recebeu o rótulo de 1.1.0, o que representa que ela foi a primeira versão evolutiva depois que o sistema entrou em produção. Essa versão adicionou ao sistema o controle de etapas e controle de prorrogações de um aluno, que foram modelados separadamente, como pode ser visto na Figura 4.9, visando à simplicidade do modelo principal. É importante notar que essa versão trouxe a correção de alguns defeitos que não necessitaram do lançamento de uma versão emergencial.

**SAPOS - Sistema de Apoio à Pós-Graduação**  
Professores >> Orientações

Professores Alunos Instituições e Cursos Configurações

Professores Orientações

Buscar Adicionar

Professor

Matrícula

Orientador Principal **Selecione uma opção** ▼

Ativas

Buscar Resetar

Professor	Matrícula	Orientador Principal	
Minerva McGonagall	DH02 - Bill Weasley	<input checked="" type="checkbox"/>	Editar Excluir Visualizar
Filius Flitwick	MH05 - Luna Lovegood	<input checked="" type="checkbox"/>	Editar Excluir Visualizar
Filius Flitwick	DH01 - Charlie Weasley	<input checked="" type="checkbox"/>	Editar Excluir Visualizar
Severus Snape	MH04 - Draco Malfoy	<input checked="" type="checkbox"/>	Editar Excluir Visualizar
Minerva McGonagall	MH02 - Hermione Granger	<input checked="" type="checkbox"/>	Editar Excluir Visualizar
Filius Flitwick	MH02 - Hermione Granger	<input type="checkbox"/>	Editar Excluir Visualizar
Alastor Moody	MH01 - Harry Potter	<input type="checkbox"/>	Editar Excluir Visualizar
Severus Snape	MH01 - Harry Potter	<input type="checkbox"/>	Editar Excluir Visualizar
Minerva McGonagall	MH01 - Harry Potter	<input type="checkbox"/>	Editar Excluir Visualizar
Albus Dumbledore	MH01 - Harry Potter	<input checked="" type="checkbox"/>	Editar Excluir Visualizar

10 Registros Encontrados

Instituto de Computação / UFF Versão 1.2.0

Figura 4.10: Busca avançada em orientações

A versão seguinte foi lançada cerca de dois meses após o lançamento da versão 1.1.0, pois esse período correspondeu a época das provas finais. A versão em questão recebeu o rótulo 1.2.0 e disponibilizou para os usuários os relatórios on-line ou buscas avançadas. A busca avançada permitiu aos usuários realizar buscas mais elaboradas sobre os registros através da utilização de diferentes parâmetros e a combinação entre eles, como pode ser visto

na Figura 4.10. No entanto, esse tipo de busca estava disponível apenas para as seções orientações e bolsas, pois, segundo os usuários, era onde esse tipo de busca seria realmente utilizado. É importante destacar que essa versão não gerou alteração no modelo de dados, mas como houve adição de novas funcionalidades, foi considerada uma versão evolutiva.

Na versão seguinte, rotulada como 1.3.0, foi disponibilizada a funcionalidade de exportar os relatórios gerados na busca avançada em formato PDF, como pode ser visto na Figura 4.11. Além disso, foi inserida a funcionalidade que calcula a quantidade de pontos de orientação por professor, baseando-se no número de alunos orientados de cada um. Por fim, foi adicionada ao sistema a possibilidade de realizar buscas sobre orientações ativas e inativas.

Universidade Federal Fluminense  
Instituto de Computação  
Pós-Graduação



Professor	Número de Matrícula	Aluno	Nível
Minerva McGonagall	DH02	Bill Weasley	Doutorado
Filius Flitwick	MH05	Luna Lovegood	Mestrado
Filius Flitwick	DH01	Charlie Weasley	Doutorado
Severus Snape	MH04	Draco Malfoy	Mestrado
Minerva McGonagall	MH02	Hermione Granger	Mestrado
Filius Flitwick	MH02	Hermione Granger	Mestrado
Alastor Moody	MH01	Harry Potter	Mestrado
Severus Snape	MH01	Harry Potter	Mestrado
Minerva McGonagall	MH01	Harry Potter	Mestrado
Albus Dumbledore	MH01	Harry Potter	Mestrado

Figura 4.11: Relatório de orientações

Dentre as mudanças trazidas pela versão 1.4.0, pode-se destacar a mudança de *layout* do sistema, como pode ser visto na Figura 4.12 (compare com o layout anterior, mostrado na Figura 4.10). Além disso, foram implementadas as funcionalidades de busca avançada e geração de relatórios em PDF para a seção Alocação de Bolsas. Com isso, todas as funcionalidades com maior grau de importância para os usuários foram incluídas no sistema.

The screenshot shows the SAPOS (Sistema de Apoio a Pós-Graduação) interface. At the top, there is a header with the SAPOS logo, the text "[ Sistema de Apoio a Pós-Graduação ]", "Instituto de Computação / UFF", and "Versão 1.4.0". A navigation menu includes "Professores", "Alunos", "Instituições e Cursos", and "Configurações". The main content area is titled "Matrículas" and features a search bar with "Buscar" and "Adicionar" buttons. Below this is a table with the following data:

	Aluno	Número de Matrícula	Nível	Tipo de Matrícula	Data de Admissão	Desligamento	
Alunos	Charlie Weasley	DH01	Doutorado	Regular	Junho-1995		Editar Excluir Visualizar
Matrículas	Bill Weasley	DH02	Doutorado	Especial	Março-1995		Editar Excluir Visualizar
	Harry Potter	MH01	Mestrado	Regular	Junho-1998		Editar Excluir Visualizar
Realização de Etapa	Hermione Granger	MH02	Mestrado	Regular	Março-1998		Editar Excluir Visualizar
	Ronald Weasley	MH03	Mestrado	Regular	Março-1998		Editar Excluir Visualizar
Bolsas	Draco Malfoy	MH04	Mestrado	Regular	Junho-1998		Editar Excluir Visualizar
	Luna Lovegood	MH05	Mestrado	Especial	Junho-1998		Editar Excluir Visualizar
Alocação de Bolsas	7 Registros Encontrados						
Prorrogações							

Figura 4.12: Novo *layout* do sistema

Por fim, a última versão do SAPOS foi responsável por mudanças de interface e reorganização dos menus, evitando a criação de um terceiro nível de navegação e visando a melhoria da usabilidade.

Além das versões citadas acima, foram criadas versões corretivas que estão distribuídas entre as versões evolutivas, por exemplo: versão 1.2.1, 1.3.2, 1.3.4, entre outras. Em números finais, foram oito mudanças evolutivas e onze mudanças corretivas.

#### 4.5.3 DESAFIOS ENCONTRADOS

O desenvolvimento do SAPOS mostrou-se interessante para o aprendizado de um novo método de desenvolvimento e uma nova linguagem de programação. Entretanto, conciliar o desenvolvimento de um sistema com a vida profissional e as demais atribuições da vida acadêmica não foi uma tarefa fácil. Em alguns períodos, foi necessário parar o desenvolvimento do SAPOS para que fosse possível desenvolver trabalhos para as demais disciplinas e estudar para as provas. Devido a isso, as versões do sistema que deveriam ser entregues ao final de cada *sprint*, eram entregues com dois ou até três *sprints* de duração.

É importante citar que os desenvolvedores deste projeto compartilhavam também de uma vida profissional. Com isso, além do conhecimento adquirido na universidade, era necessário estudar além do horário de trabalho para manter o rendimento no mesmo. Assim como a vida acadêmica, a vida profissional também influenciou nos *sprints*, atrasando-os em períodos de maior exigência.

Por fim, é importante ressaltar a etapa de análise e migração dos dados registrados nos antigos bancos de dados da Pós-Graduação. Esta etapa, com cerca de dois meses de duração, estendeu-se mais do que o planejado devido a complexidade de migração dos dados do *Microsoft Access* para o MySQL, como explicitado na Seção 4.4.

#### **4.6 CONSIDREÇÕES FINAIS**

Neste capítulo foram apresentadas as técnicas e artefatos utilizados no desenvolvimento do SAPOS. Através do método adotado, procurou-se desenvolver um sistema que atendesse às necessidades apresentadas pela coordenação da Pós-Graduação do IC-UFF. Utilizando uma linguagem de programação alinhada às técnicas ágeis, foi possível entregar um software funcionando a cada versão do sistema. Por fim, os mecanismos existentes na linguagem permitiram a realização da migração de dados de forma segura e eficaz.

## CAPÍTULO 5 – CONCLUSÃO

O resultado deste trabalho consiste na criação de um sistema de gerência de dados para a Pós-Graduação do IC-UFF. Sua principal contribuição é a organização dos dados manipulados pela Pós-Graduação, facilitando o registro e a visualização das informações armazenadas. Com o SAPOS, passou a ser possível centralizar as informações até então gerenciadas através de diversas planilhas *Microsoft Excel* e de bancos de dados *Microsoft Access*, possibilitando a criação de uma base de dados única.

Os trabalhos relacionados apresentados neste projeto tratam, em sua maioria, da gestão acadêmica dos alunos da Pós-Graduação. Em comparação com esses sistemas, pode-se dizer que o SAPOS atua na gerência dos dados acadêmico-administrativos da Pós-Graduação, tratando tanto dos dados acadêmicos dos alunos como dos dados administrativos (controle de bolsas de fomento, pontos de orientação, entre outros), manipulados pela secretaria e coordenação da Pós-Graduação.

Fazendo uma análise crítica sobre o sistema, é possível citar alguns pontos que precisam de melhoria. Como exemplo, pode-se citar o controle de etapas concluídas por um aluno, pois, da maneira como foi projetada, essa funcionalidade considera que períodos letivos possuem o mesmo número de meses. Dessa forma, quando o número de meses em cada período é diferente, faz-se necessária a criação de uma etapa para cada semestre, como “prova de inglês – 1º semestre” e “prova de inglês – 2º semestre”. Assim, o usuário deve ficar atento para: (i) ao cadastrar uma nova etapa, criar um registro para o primeiro e um para o segundo semestre, e (ii) ao atribuir a conclusão de uma etapa a um aluno, escolher o semestre que o aluno concluiu a determinada etapa.

Na atual versão do SAPOS não existem alertas, ou seja, notificações que consistem em mensagens enviadas em forma de correio eletrônico ou janelas dentro do sistema com a finalidade de avisar o usuário sobre alguma situação. Devido a isso, diversas informações registradas no sistema estão sendo subutilizadas. Pode-se citar novamente o controle de etapas, onde cada aluno tem cadastrado a data de admissão e o prazo máximo para o cumprimento de uma etapa. Contudo, o sistema não gera nenhum tipo de alerta para o aluno solicitar prorrogação. Outro exemplo está no registro de pontos de orientação, pois, ao associar um aluno a um professor, o sistema não gera nenhum tipo de alerta caso o professor já esteja orientando o número máximo de alunos.

Ainda na versão atual do sistema, pode-se citar a geração de relatórios como ponto que necessita de melhoria, pois é possível gerar relatórios em formato PDF apenas nas seções

orientações e alocação de bolsas, localizadas nas áreas Professores e Bolsas, respectivamente. Além disso, nos relatórios constam apenas as informações previamente definidas na tela de visualização inicial de cada seção do sistema (Orientações, Alocação de Bolsas, Cursos, entre outras). Apesar da consulta poder ser realizada utilizando uma combinação dos dados manipulados em cada seção (Busca avançada), o modo como o relatório foi implementado impede que sejam gerados documentos com a estrutura personalizável. Apesar das carências citadas acima, o SAPOS atende às necessidades prioritárias da Pós-Graduação do IC-UFF e já está sendo utilizado pelas secretárias da Pós-Graduação, substituindo totalmente as planilhas utilizadas anteriormente (SILVA *et al.*, 2012).

No entanto, além dos pontos citados, existem outros aspectos que estão sendo melhorados pelos alunos Bruno Schettino (aluno de Ciência da Computação da UFF) e Everton Moreth (aluno de Sistemas de Informação da UFF), responsáveis pela continuação do SAPOS. Dentre as melhorias que estão sendo desenvolvidas no sistema, pode-se destacar o aprimoramento do controle de acesso. No momento, o SAPOS conta com uma autenticação via login, onde todos os usuários possuem acesso a todas as funcionalidades do sistema. Entretanto, a evolução do controle de acesso visa segregar os tipos de usuário: alunos, professores, secretaria, coordenação e administrador, permitindo, assim, que algumas áreas do sistema sejam acessadas apenas por determinados tipos de usuários. Além disso, cada perfil terá permissões de leitura e escrita dentro da área do sistema a que possui acesso. Adicionalmente, para facilitar o acesso dos usuários, está sendo desenvolvida a funcionalidade de “Esqueci minha senha”, que é responsável por enviar ao usuário solicitante uma nova senha.

Outra melhoria a ser desenvolvida é a criação de um ambiente de homologação. Esse ambiente é um clone do ambiente de produção. No entanto, serve para validação do que foi desenvolvido com os usuários finais. O novo ambiente permitirá a automatização do processo de cópia do banco de dados, possibilitando que os desenvolvedores disponham de uma base com informações atualizadas, importante instrumento para a execução de testes. Ao mesmo tempo, isola os desenvolvedores do ambiente de produção, que fica acessível apenas aos usuários autorizados.

Os pontos levantados fazem referência ao aprimoramento de requisitos não funcionais. Entretanto, a adição constante de funcionalidades é essencial para que o sistema continue atendendo as necessidades da Pós-Graduação. Pode-se citar como próximas funcionalidades a serem desenvolvidas: o controle de disciplinas, controle de notas e controle de áreas de pesquisa.

A ideia de inserir um controle sobre as disciplinas cursadas pelos alunos da Pós-Graduação vem do desejo de acabar com os controles paralelos que ainda vigoram na secretaria. Dados como disciplinas cursadas, professores que as ministram e áreas de pesquisa seriam disponibilizados no histórico do aluno, permitindo, assim, que possa ser feito um controle sobre que disciplinas ainda devem ser cumpridas por um aluno.

Juntamente com o controle de disciplinas, tem-se a ideia de criar o controle de notas. Essa funcionalidade permitirá que os professores atribuam uma nota a um aluno matriculado em uma turma. Com a implementação do controle de notas, o SAPOS conseguirá substituir o banco de dados *Microsoft Access* que ainda vem sendo utilizado unicamente para esse fim, trazendo maior consistência às informações manipuladas na secretaria da Pós-Graduação do IC-UFF.



## REFERÊNCIAS BIBLIOGRÁFICAS

AVILA, CHRISTIANO. *Projeto de Fim de Curso (UFF)*. [mensagem informativa]. Mensagem recebida por: tiagoamaro@id.uff.br em: 8 ago. 2012

BORGES, MARCELI. *Projeto de Fim de Curso (UFF)*. [mensagem informativa]. Mensagem recebida por: tiagoamaro@id.uff.br em 12 dez. 2012.

BRAGANHOLO, VANESSA. *Números sobre a pós #2*. [mensagem informativa]. Mensagem recebida por: tiagocis@gmail.com em 1 out. 2012.

COHEN, D.; LINDVALL, M.; COSTA, P. An Introduction to Agile Methods. *Advances in Computers*. [S.l.]: Elsevier, 2004. v. Volume 62. p. 1–66. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0065245803620012>>. Acesso em: 12 fev. 2013.

FIELDING, R. T. *Architectural Styles and the Design of Network-based Software Architectures*. 2000. 162 f. University of California, California, Irvine, 2000. Disponível em: <[http://www.ics.uci.edu/~fielding/pubs/dissertation/fielding\\_dissertation.pdf](http://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf)>. Acesso em: 3 set. 2012.

FLANAGAN, D.; MATSUMOTO, Y. *The Ruby Programming Language*. First Edition ed. [S.l.]: O'Reilly Media, 2008.

GREGORY, B.; HEALY, J. *Prawn*. Disponível em: <<http://prawn.majesticseacreature.com/>>. Acesso em: 3 set. 2012.

HANSSON, D. *Action Pack — On rails from request to response*. Disponível em: <<http://ap.rubyonrails.org/>>. Acesso em: 3 set. 2012a.

HANSSON, D. *ActiveRecord – Object-relational mapping put on rails*. Disponível em: <<http://ar.rubyonrails.org/>>. Acesso em: 3 set. 2012b.

HANSSON, D. *db/seeds.rb added as default*. Repositório. Disponível em: <<https://github.com/rails/rails/commit/f3c7bbeedd81d2f379c5e6a9e8739d3b3784ca5f>>. Acesso em: 10 set. 2012.

JETBRAINS. *RubyMine*. Disponível em: <<http://www.jetbrains.com/ruby/>>. Acesso em: 3 nov. 2012.

KERNIGHAN, B. W.; PIKE, R. *Unix Programming Environment*. [S.l.]: Prentice Hall, 1984.

KRASNER, Glenn E.; POPE, Stephen. A Description of the Model-View-Controller User Interface Paradigm in the Smalltalk-80 System. 1988. Disponível em: <<http://www.create.ucsb.edu/~stp/PostScript/mvc.pdf>>. Acesso em: 3 set. 2012.

LANG, J.-P. *Redmine*. Disponível em: <<http://www.redmine.org/>>. Acesso em: 23 out. 2012.

MATSUMOTO, Y. *Ruby Programming Language*. Disponível em: <<http://www.ruby-lang.org/en/about/>>. Acesso em: 3 set. 2012.

MEEHAN, A. *validates\_timeliness*. Disponível em: <[https://github.com/adzap/validates\\_timeliness](https://github.com/adzap/validates_timeliness)>. Acesso em: 3 set. 2012.

*Net Beans Community News*. Disponível em: <<http://netbeans.org/community/news/show/1507.html>>. Acesso em: 3 nov. 2012.

ORACLE. *NetBeans*. Disponível em: <<http://netbeans.org/>>. Acesso em: 3 nov. 2012.

PGC. *Regras de Prorrogação e Prazos Máximos*. Disponível em: <[http://www.ic.uff.br/PosGraduacao/regras\\_e\\_procedimentos\\_arquivos/regras\\_de\\_prorrogacao\\_e\\_prazos\\_maximos\\_v2.pdf](http://www.ic.uff.br/PosGraduacao/regras_e_procedimentos_arquivos/regras_de_prorrogacao_e_prazos_maximos_v2.pdf)>. Acesso em: 29 ago. 2012.

PHUSION. *Phusion Passenger*. Disponível em: <<https://www.phusionpassenger.com/>>. Acesso em: 3 nov. 2012a.

PHUSION. *Phusion Passenger users guide*. Disponível em: <[http://www.modrails.com/documentation/Users%20guide%20Apache.html#\\_redeploying\\_restarting\\_the\\_ruby\\_on\\_rails\\_application](http://www.modrails.com/documentation/Users%20guide%20Apache.html#_redeploying_restarting_the_ruby_on_rails_application)>. Acesso em: 3 nov. 2012b.

PRESSMAN, R. S. *Software Engineering: A Practitioner's Approach*. [S.l.]: McGraw-Hill, 2004.

QUARANTO, N. *RubyGems.org*. Disponível em: <<http://rubygems.org/>>. Acesso em: 25 fev. 2013.

REZENDE, D. *Engenharia de Software e Sistemas de Informação*. [S.l.]: Brasport, 2005.

RIBEIRO, C. *Ata da Reunião do Colegiado da Pós-Graduação*. [S.l.: s.n.], 25 maio 2011.

RUBY, S.; THOMAS, D.; HANSSON, D. H. *Agile Web Development with Rails*. Fourth Edition ed. [S.l.]: Pragmatic Bookshelf, 2011.

RUTHERFORD, S. *ActiveScaffold :: A Ruby on Rails plugin from the makers of AjaxScaffold*. Disponível em: <[https://github.com/activescaffold/active\\_scaffold](https://github.com/activescaffold/active_scaffold)>. Acesso em: 20 jan. 2013.

SCHWABER, K. *Agile Project Management with Scrum*. 1. ed. [S.l.]: Microsoft Press, 2004.

SILVA, E. *et al. InformeIC - Informativo do Instituto de Computação da UFF*. [S.l.: s.n.]. Disponível em: <<http://www.youblisher.com/p/501211-InformeIC-Dezembro-de-2012/>>. Acesso em: 20 jan. 2013. , 2012

SORENSEN, J. *Gitorious*. Disponível em: <<http://gitorious.org/>>. Acesso em: 3 nov. 2012.

TIOBE. *TIOBE Programming Community Index for January 2013*. Disponível em: <<http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html>>. Acesso em: 9 jan. 2013.

TORVALDS, L. *Git*. Disponível em: <<http://git-scm.com/>>. Acesso em: 23 out. 2012.

VALENTIM, M. *Gestão, mediação e uso da informação*. [S.l.]: UNESP, 2010.

WANSTRATH, C.; HYETT, P.; PRESTON-WERNER, T. *GitHub*. Disponível em: <<https://github.com/>>. Acesso em: 3 nov. 2012.

YLONEN, T.; LONVICK, C. *The Secure Shell (SSH) Authentication Protocol*. Disponível em: <<http://tools.ietf.org/html/rfc4252>>. Acesso em: 3 nov. 2012.