

UNIVERSIDADE FEDERAL FLUMINENSE  
INSTITUTO DE COMPUTAÇÃO

João Felipe Nicolaci Pimentel  
Luis Antônio Vieira Junior

SAPOS 3: ADIÇÃO DE NOTIFICAÇÕES AO  
SISTEMA DE APOIO A PÓS-GRADUAÇÃO

Niterói  
2014

JOÃO FELIPE NICOLACI PIMENTEL  
LUIS ANTÔNIO VIEIRA JUNIOR

SAPOS 3: ADIÇÃO DE NOTIFICAÇÕES AO SISTEMA DE APOIO A PÓS-  
GRADUAÇÃO

Monografia apresentada ao Curso de  
Graduação em Ciência da Computação da  
Universidade Federal Fluminense, como  
requisito parcial para obtenção do Grau  
de Bacharel em Ciência da Computação.

Orientadores: Prof. Dr. Leonardo Gresta Paulino Murta  
Prof. Dr. Vanessa Braganholo Murta

Niterói  
2014

JOÃO FELIPE NICOLACI PIMENTEL

LUIS ANTÔNIO VIEIRA JUNIOR

SAPOS 3: ADIÇÃO DE NOTIFICAÇÕES AO SISTEMA DE APOIO A PÓS-  
GRADUAÇÃO

Monografia apresentada ao Curso de  
Graduação em Ciência da Computação da  
Universidade Federal Fluminense, como  
requisito parcial para obtenção do Grau  
de Bacharel em Ciência da Computação.

Aprovada em maio de 2014.

BANCA EXAMINADORA

---

Prof. Dr. LEONARDO GRESTA PAULINO MURTA – Orientador  
UFF

---

Prof. Dr. VANESSA BRAGANHOLO MURTA – Orientador  
UFF

---

Prof. Dr. ESTEBAN WALTER GONZALEZ CLUA  
UFF

---

Prof. Dr. FÁBIO PROTTI  
UFF

Niterói

2014

## AGRADECIMENTOS (JOÃO FELIPE)

Agradeço, primeiramente, aos meus pais: Rui Alberto e Angela Maria, por todo apoio e investimento, educação, amor, carinho, incentivo e exemplo que me deram ao longo de toda a minha vida.

Ao meu irmão mais velho, Carlos Augusto, por ter me ensinado a programar, me motivando a seguir este caminho.

Ao meu irmão mais novo, Marcelo, por toda companhia, por ter ajudado a me distrair com jogos e filmes no tempo livre.

Agradeço ao resto da minha família por todo incentivo e carinho.

À Universidade Federal Fluminense por ter disponibilizado toda a estrutura necessária para o aprendizado.

Ao Instituto de Computação por todo apoio e suporte na realização das atividades acadêmicas e até mesmo na realização de atividades extracurriculares, como Dojo e Infomarka.

Aos ótimos professores da UFF, que sempre se esforçaram para dar boas aulas e passar o conhecimento. E também aos professores razoáveis, por mostrarem que livros também devem ser usados na busca do conhecimento.

Ao professor Otton Teixeira da Silveira Filho, por ter me aconselhado a entrar na UFF e não em outras universidades.

Ao professor Leonardo Gresta Paulino Murta por ter me apresentado o mundo das pesquisas, como meu orientador de Iniciação Científica de 2010 a 2012. Agradeço também ao professor Christiano de Oliveira Braga por ter me recomendado ao Leo.

A todos os professores e amigos que me incentivaram a participar do Ciência Sem Fronteiras.

À *Washington University* in St. Louis (WashU), com seus ótimos professores, que me aceitou como intercambista e me fez ter uma visão cultural diferente do mundo.

Às orientadoras da WashU, Melanie Osborn e Kaaren Quezada, que ajudaram com os mais diversos assuntos, desde moradia até sugestão de como me inscrever em uma turma lotada.

A todos os amigos que fiz em *St. Louis* que me permitiram conhecer a cultura de diversos lugares do Brasil e do mundo e que colaboraram para que eu tivesse o melhor ano da minha vida.

Agradeço novamente ao professor Leonardo Gresta Paulino Murta e à professora Vanessa Braganholo Murta por me aceitarem como orientando do projeto final e por todo suporte e apoio que conseguiram dar em qualquer problema encontrado, mesmo quando estavam muito distantes fisicamente.

Ao meu amigo Luis Antônio Vieira Junior, por ter participado deste projeto e de muitos outros trabalhos comigo e por todo convívio desde que entramos na UFF.

A todos os meus amigos da faculdade. Principalmente aos que ingressaram comigo em 2010.1 e aos que se juntaram ao grupo posteriormente, por todo o conhecimento compartilhado nos grupos de estudo, toda a convivência e companheirismo durante todo este tempo.

Ao pessoal do pós-dojo por todas as quintas-feiras e por me mostrarem que também existe vida fora da universidade.

A todos os outros que contribuíram de alguma forma para a minha formação, seja pessoal ou acadêmica.

Muito obrigado!

## AGRADECIMENTOS (LUIS ANTÔNIO)

Agradeço à minha família, em especial minha mãe Sandra Regina Freitas Silva Vieira e meu pai Luis Antônio Vieira, por todo o investimento, não só financeiro, mas de uma vida inteira, em prol da minha formação e educação. Agradeço a eles também por todo o amor, carinho e incentivo ao longo desses mais de 22 anos, por terem lutado para que eu estudasse em uma boa escola e com isso tivesse condições de estudar em uma excelente universidade que é a Universidade Federal Fluminense. Por terem me ensinado a me esforçar e lutar pelo que eu quero, mostrando que tudo na vida é recompensa do meu esforço e que eu mesmo deveria buscar os meus objetivos, sem precisar para isso prejudicar ninguém ao meu redor, sem vocês e os demais da família eu dificilmente teria chegado a este ponto, por isso eu lhes agradeço com toda a minha sinceridade.

Aos professores Leonardo Gresta Paulino Murta e Vanessa Braganholo Murta, primeiramente por me aceitarem como seu orientando, me possibilitando a chance de demonstrar minha dedicação e compromisso, e também por todo apoio dado a mim e ao João durante este projeto, e que mesmo estando a milhares de quilômetros de distância de nós durante grande parte do processo, nunca hesitaram em nos auxiliar e estavam sempre prontos para responder qualquer dúvida e ajudar em qualquer problema que tivemos.

A meu grande amigo João Felipe Nicolaci Pimentel, que aceitou participar deste projeto comigo e que trouxe uma grande qualidade técnica ao mesmo, mas não só por isso, também por todos os anos de convívio desde março de 2010 quando ingressamos juntos na Universidade Federal Fluminense.

Aos meus amigos do Bacharelado em Ciência da Computação da Universidade Federal Fluminense, em especial os que ingressaram comigo no primeiro semestre de 2010, que ao longo destes anos me ajudaram, não apenas academicamente mas também fizeram parte da construção de quem eu sou hoje, e a todos os outros amigos da UFF que de alguma maneira estiveram comigo nessa jornada.

A todos que estiveram comigo e me ajudaram em um dos anos mais importantes da minha vida, enquanto estive na *Colorado School of Mines* durante 2012.

Agradeço à Dr. Kay Godel-Gengenbach e Teuku Andika, do *Office of International Programs* da *Colorado School of Mines*, que desde antes de minha partida do Brasil para o Colorado, já estavam de prontidão para me ajudar em qualquer situação.

Um agradecimento muito especial a Lorna Crawford, que se tornou uma mãe para todos nós Cientistas Sem Fronteiras na *Colorado School of Mines*.

Aos amigos Cientistas Sem Fronteiras que estiveram comigo durante este ano, vocês foram a peça chave nesta jornada.

A todos os amigos que fiz em 2012 enquanto estive na *Colorado School of Mines*, especialmente meus *floormates* do *Randall Basement*, minha família no Colorado, para mim vocês foram a parte principal deste ano. Fico muito feliz de vocês fazerem parte da minha vida, sem vocês tudo teria sido mais difícil. Muito obrigado por terem me aceitado na vida de vocês apesar de todas as diferenças.

Agradeço a UFF por me proporcionar um grande aprendizado e experiência, além de me capacitar para ser um grande Cientista da Computação.

Por fim, agradeço a todos os demais que de alguma forma contribuíram para a minha formação, seja pessoal ou acadêmica, meus sinceros agradecimentos. Muito obrigado!

## RESUMO

O Sistema de Apoio à Pós-Graduação (SAPOS) foi criado para suprir as necessidades da coordenação dos cursos de Pós-Graduação do Instituto de Computação da Universidade Federal Fluminense (IC-UFF). Em sua segunda versão o sistema supria necessidades de gerenciamento de dados referentes a alunos, matrículas, professores, credenciamentos, orientações, bolsas de fomento, realização de etapas obrigatórias, disciplinas e de outras informações relacionadas aos cursos de pós-graduação.

Entretanto, uma das necessidades da coordenação que a segunda versão do SAPOS não tratava era o envio de notificações para coordenadores, orientadores, professores, secretaria, e até mesmo para os alunos para lembrar de prazos para realização de etapas; avisar o orientador sobre novos orientandos e notas de seus orientandos e outras possíveis notificações. Este trabalho, a terceira versão do SAPOS, consiste na concepção e implementação de um sistema de notificações configurável para suprir essas necessidades e na implementação de diversas manutenções corretivas e funcionalidades menores, tais como remodelar o histórico do aluno para o formato definido pela Pró-Reitoria de Pesquisa, Pós-graduação e Inovação da UFF (PROPPi), adicionar *log* de todas as operações realizadas no sistema, melhorar cadastro de endereços, entre outras. O SAPOS é um projeto *web* desenvolvido em *Ruby on Rails*, um *framework web* para linguagem de programação *Ruby*.

Palavras-chave: Notificações, Gestão Acadêmica de Pós-Graduação.



## **ABSTRACT**

The Graduate Support System (SAPOS) was created to meet the needs of the Graduate Courses Coordination of the Computing Institute at Fluminense Federal University (IC-UFF). In its second version, the system was able to manage data referring to students, enrollments, professors, accreditations, orientations, scholarships, achievements on mandatory phases, courses, and other information related to graduate studies. However, one of the needs of the coordination, that the second version of SAPOS did not meet, was sending notifications to coordinators, advisors, professors, staff and even to the students. These notifications help on reminding them of due dates to some phases, warning advisors of new advisees and grades of their advisees, among other possibilities. This project, the third version of SAPOS, consists on the conception and implementation of a configurable notification system to meet these needs, some corrective maintenance, and minor features, such as remodeling the student report card to the format required by the high level administration of the university, adding log to every operation made on the system, and improving the address record, amongst others.

This is a web project, developed in Ruby on Rails, a web framework for the Ruby programming language.

**Keywords:** Notifications, Graduate Academic Management.

## SUMÁRIO

Capítulo 1 – Introdução .....	15
Capítulo 2 – Visão Geral do SAPOS.....	17
2.1 SAPOS 1 .....	17
2.1.1 Alunos.....	17
2.1.2 Professores .....	18
2.1.3 Bolsas.....	19
2.1.4 Etapas.....	19
2.1.5 Formação .....	20
2.1.6 Localidades .....	21
2.1.7 Configurações .....	21
2.2 SAPOS 2 .....	21
2.2.1 Novo controle de etapas e prorrogações .....	21
2.2.2 Busca avançada de matrículas .....	22
2.2.3 Controle de credenciamentos.....	23
2.2.4 Controle de disciplinas .....	23
2.2.5 Novos relatórios do SAPOS 2 .....	24
2.3 Discussão .....	26
Capítulo 3 – SAPOS 3: Notificações.....	27
3.1 Notificações Programadas .....	27
3.1.1 Agendamento de Notificações .....	30
3.1.2 Realização de Consultas .....	33
3.1.3 Envio de Mensagens .....	37
3.1.4 Cadastro de Notificações Programadas .....	40
3.2 Notificações Fixas.....	41
3.3 Discussão .....	42

Capítulo 4 – SAPOS 3: Outras Melhorias .....	44
4.1 Sistema de Registro de Mudanças .....	44
4.2 Variáveis Customizáveis.....	45
4.3 Geração de Relatórios .....	47
4.4 Melhoria na Visualização e Interação.....	51
4.5 Discussão .....	54
Capítulo 5 – Conclusão .....	56
Referências Bibliográficas.....	57
Apêndice A – Alterações no Modelo de Classes do SAPOS 3 em Relação ao SAPOS 2 .....	59
Anexo A - Notificações Programadas Cadastradas .....	60

## LISTA DE ILUSTRAÇÕES

Figura 2.1: Tela inicial do sistema (matrículas) .....	18
Figura 2.2: Tela de professores.....	19
Figura 2.3: Tela de realização de etapas .....	20
Figura 2.4: Tela de formação .....	20
Figura 2.5: Tela de localidades (estados) .....	21
Figura 2.6: Tela de configurações.....	21
Figura 2.7: Busca avançada de matrículas.....	22
Figura 2.8: Tela de credenciamentos .....	23
Figura 2.9: Tela de disciplinas .....	24
Figura 2.10: Pauta de turma.....	25
Figura 2.11: Histórico escolar.....	25
Figura 3.1: Cadastro de Notificação Programada.....	29
Figura 3.2: Frequência ajustada por deslocamento.....	30
Figura 3.3: Agendamento de Notificações Programadas.....	32
Figura 3.4: Consulta de matrículas ativas .....	34
Figura 3.5: Consulta do número de alunos ativos.....	34
Figura 3.6: Frequência ajustada por dois deslocamentos .....	35
Figura 3.7: Exemplo de modelo para e-mail único.....	38
Figura 3.8: Exemplo de modelo para e-mail para cada tupla .....	39
Figura 3.9: Página de Notificações .....	41
Figura 3.10: <i>Links</i> da Página de Notificações .....	41
Figura 4.1: Tela de Registro de Mudanças .....	45
Figura 4.2: Relatório de quadro de horários .....	48
Figura 4.3: Histórico extraído do SAPOS 3 .....	49
Figura 4.4: Boletim extraído do SAPOS 3 .....	50
Figura 4.5: Última página da pauta extraída do SAPOS 3 .....	51
Figura 4.6: Relatório de matrículas extraído do SAPOS 3 .....	51
Figura 4.7: Tabela de matrículas .....	52
Figura 4.8: Campo Local de Expedição.....	53
Figura 4.9: Visualização de Bolsa .....	54
Figura 4.10: Página de Prorrogações .....	54

## LISTA DE TABELAS

Tabela 3.1: Data de execução base para cada frequência .....	31
Tabela 3.2: Relação de unidades para deslocamento.....	31
Tabela 3.3: Variáveis de consulta .....	36
Tabela 3.4: <i>Tags</i> do <i>ERB</i> .....	37
Tabela 3.5: Formatos para localize .....	39
Tabela 4.1: Variáveis implementadas no sistema.....	46

## LISTA DE ABREVIATURAS E SIGLAS

CAPES: Coordenação de Aperfeiçoamento de Pessoal de Nível Superior

CPF: Cadastro de Pessoas Físicas

ERB: *Embedded Ruby*

IC-UFF: Instituto de Computação da Universidade Federal Fluminense

PDF: *Portable Document Format*

PGC: Programa de Pós-Graduação em Computação

PROPPi: Pró-Reitoria de Pesquisa, Pós-Graduação e Inovação

SAPOS: Sistema de Apoio à Pós-Graduação

SQL: *Structured Query Language*

TCC: Trabalho de Conclusão de Curso

UFF: Universidade Federal Fluminense

URL: *Uniform Resource Locator*

## CAPÍTULO 1 – INTRODUÇÃO

Antes da existência do Sistema de Apoio à Pós-Graduação, a coordenação da Pós-Graduação em Computação da UFF armazenava seus dados de forma espalhada em arquivos Microsoft Excel e Microsoft Access, tornando difíceis as consultas e a manutenção dos dados (FERREIRA; AMARO, 2013). Isso ocorria pois era difícil saber em quais arquivos estavam as informações e muitas vezes era necessário atualizar uma mesma informação em mais de um local para manter a consistência e integridade.

Por conta disso, o SAPOS foi desenvolvido como um projeto web em Ruby on Rails para suprir as necessidades de armazenamento, organização e gerenciamento dos dados da Pós-Graduação em Computação da UFF. A implantação deste sistema solucionou grande parte do problema de gerenciamento de dados e, conseqüentemente, gerou novas necessidades.

Uma das novas necessidades que surgiram com a implantação do SAPOS foi a criação de relatórios, tais como histórico, boletim, quadro de horários, entre outros. Antes do SAPOS, para se gerar o histórico para um aluno, era necessário preencher um documento com as informações do aluno, as disciplinas em que ele foi aprovado, e, para cada disciplina, preencher a nota obtida e a carga horária, entre outras informações. Com o SAPOS e a centralização dessas informações, passou a ser possível a geração automática do histórico e de outros relatórios.

Outra necessidade que surgiu com a centralização de dados foi o envio de notificações para alertar coordenadores, orientadores, professores, secretaria e até mesmo alunos para lembrar de prazos para realização de etapas. Além disso, também vislumbrou-se a possibilidade de avisar orientadores sobre novos orientandos e notas de seus orientandos, apresentar periodicamente relatórios para coordenadores, entre outras possíveis notificações.

Para garantir a segurança dos dados e a confiabilidade na edição, o armazenamento de registros de mudanças (log) também foi uma das necessidades que surgiram. Este trabalho, a terceira versão do SAPOS, foi desenvolvido com o objetivo de atender às necessidades supracitadas, de notificações, registros de mudanças, geração de relatórios, entre outras. Parte da necessidade da geração de relatórios foi atendida na segunda versão do SAPOS (SCHETTINO, 2013), entretanto ainda havia a necessidade de adaptar o histórico para o formato aprovado pela Pró-Reitoria de Pesquisa, Pós-

Graduação e Inovação da UFF (PROPPi) e criar a o sistema de geração de quadro de horários.

O resto deste trabalho está dividido em quatro capítulos além da introdução. O Capítulo 2 apresenta a história do SAPOS, mostrando o que as versões anteriores disponibilizaram. O Capítulo 3 apresenta o sistema de notificações, explicando como este foi implementado, a diferença entre notificações fixas e programadas e como ocorre o cadastro de uma nova notificação. O Capítulo 4 apresenta as outras melhorias que foram feitas no sistema, tais como o sistema de registro de mudanças, as alterações no sistema de geração de relatórios, a introdução de variáveis customizáveis e as melhorias na visualização e interação. Por fim, o Capítulo 5 conclui o trabalho, discutindo os objetivos alcançados e apresentando possíveis melhorias que podem ser realizadas no futuro.



## CAPÍTULO 2 – VISÃO GERAL DO SAPOS

Devido ao aumento do número de docentes credenciados e alunos matriculados no Programa de Pós-Graduação em Computação (PGC) e a grande quantidade de sistemas e ferramentas utilizados pela secretaria do PGC para gerenciar estes dados, surgiu a necessidade de se desenvolver um sistema único e centralizado, que se tornaria responsável por organizar e gerenciar todos os dados do PGC. Tendo em vista esta necessidade, o coordenador do PGC, Prof. Celso Ribeiro procurou o Prof. Leonardo Murta, que, junto com a Prof. Vanessa Braganholo, criou um protótipo inicial do SAPOS. A partir desta versão inicial, o SAPOS foi tema de dois Trabalho de Conclusão de Curso (TCC) anteriores a este TCC.

Neste capítulo apresentamos as evoluções que estes dois TCCs trouxeram ao SAPOS. O SAPOS 1, resultado do TCC dos alunos Rodrigo Dias Ferreira e Tiago Manuel Padrela de Amaro (2013) e o SAPOS 2, resultado do TCC do aluno Bruno de Pinho Schettino (2013).

### 2.1 SAPOS 1

Em sua primeira versão, o SAPOS contemplava o cadastro de informações de alunos e suas matrículas, professores e suas orientações, bolsas e etapas. Um total de sete seções principais foram criadas: alunos, professores, bolsas, etapas, formação, localidades e configurações. Cada uma destas áreas foi subdividida em subseções, que continham informações pertinentes a estas, como, por exemplo, a área de Professores, que foi subdividida em professores e orientações.

#### 2.1.1 ALUNOS

A área de alunos é a principal do sistema, onde ficam concentradas todas as informações sobre os discentes do PGC. Por sua importância foi escolhida como página inicial do sistema após o *login*. A área de Alunos é subdividida em seis diferentes subseções:

**Alunos** – Seção que descreve as características do discente, como nome, CPF e logradouro.

**Matrículas** – Seção que relaciona um aluno a um nível de curso e contém todas as informações deste aluno em relação a este curso. Uma matrícula está relacionada a um aluno, mas um aluno pode ter mais de uma matrícula (por exemplo, uma no mestrado e

outra, mais tarde, no doutorado). É considerada a área central do sistema. A Figura 2.1 apresenta a tela de matrículas.

**Níveis** – Seção onde estão cadastrados os níveis do curso (e.g., especialização, mestrado e doutorado).

**Tipos de Matrícula** – Seção onde estão cadastradas as possíveis ligações que uma matrícula tem ao curso (e.g., avulso ou regular).

**Razões de Desligamento** – Seção onde estão cadastradas as possíveis razões de desligamento de uma matrícula (e.g., desistência, prazo e titulação).

**Desligamentos** – Seção que se refere às matrículas que por algum motivo foram desligadas do curso.

	Aluno	Número de Matrícula	Nível	Tipo de Matrícula	Data de Admissão	Desligamento	
Alunos	Charlie Weasley	DH01	Doutorado	Regular	Junho-1995		Editar Excluir Visualizar
	Bill Weasley	DH02	Doutorado	Especial	Março-1995		Editar Excluir Visualizar
Desligamentos	Harry Potter	MH01	Mestrado	Regular	Junho-1998		Editar Excluir Visualizar
Matrículas	Hermione Granger	MH02	Mestrado	Regular	Março-1996		Editar Excluir Visualizar
	Ronald Weasley	MH03	Mestrado	Regular	Março-1998		Editar Excluir Visualizar
Níveis	Draco Malfoy	MH04	Mestrado	Regular	Junho-1996		Editar Excluir Visualizar
	Luna Lovegood	MH05	Mestrado	Especial	Junho-1998		Editar Excluir Visualizar
Razões de Desligamento	7 Registros Encontrados						
Tipos de Matrícula							

**Figura 2.1: Tela inicial do sistema (matrículas)**

## 2.1.2 PROFESSORES

A área de professores contempla o armazenamento de informações básicas dos professores e de suas relações com os alunos (i.e., orientações). Esta área é subdividida em duas subseções:

**Professores** – Seção que descreve as características do professor, como nome, CPF, logradouro e pontos de orientação. A pontuação de um professor é calculada através de suas orientações. Para cada orientando, um ponto é somado caso o professor seja o único orientador credenciado do aluno. Caso ele coorientar o aluno com outro professor credenciado, apenas meio ponto é somado. De acordo com as regras em vigor no PGC, um docente não pode ter mais do que oito pontos de orientação. A Figura 2.2 apresenta a tela de professores.

**Orientações** – Seção que relaciona uma matrícula de aluno a um ou mais professores, que são os orientadores/coorientadores deste aluno.

Alunos

Professores

Bolsas

Etapas

Formação

Localidades

Configurações

Professores

Buscar

Adicionar

Professores

Orientações

Nome

Alastor Moody

CPF

22222222229

Data de Nascimento

-

Pontos de Orientação

0.5

Editar

Excluir

Visualizar

Albus Dumbledore

22222222219

-

0.5

Editar

Excluir

Visualizar

Nome

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

Siapa

-

1º Telefone

-

2º Telefone

-

CEP

-

Bolsas

PH02

Orientandos

DH01 - Charlie Weasley, MH05 - Luna Lovegood, MH02 - Hermione Granger

Fechar

Minerva McGonagall

22222222221

-

1.5

Editar

Excluir

Visualizar

Poppy Pomfrey

22222222230

-

0.0

Editar

Excluir

Visualizar

Remus Lupin

22222222222

-

0.0

Editar

Excluir

Visualizar

Severus Snape

22222222223

-

2.0

Editar

Excluir

Visualizar

7 Registros Encontrados

**Figura 2.2: Tela de professores**

### 2.1.3 BOLSAS

A área de bolsas armazena informações essenciais ao gerenciamento diário das bolsas de fomento, como número de identificação, agência de fomento, data de início e fim, nível, entre outras. Esta área também é responsável por relacionar um aluno a uma bolsa através de sua matrícula, caso este aluno tenha recebido uma bolsa. Essa relação é chamada de “alocação”.

A alocação de bolsas é feita relacionando uma bolsa à uma matrícula. Durante o ciclo de vida de uma bolsa ela pode ser alocada a diversas matrículas. Uma alocação contém 3 datas importantes: data de início, data de fim, e a data limite, que determina o prazo máximo que esta bolsa pode ficar alocada àquela matrícula.

### 2.1.4 ETAPAS

A área de etapas contempla as etapas que os alunos são obrigados a completar durante o curso de Pós-Graduação para receber sua titulação, como prova de inglês, pedido de banca, entre outras. Esta é uma área muito importante administrativamente, pois permite que a secretaria ou a coordenação analise quando um aluno está próximo de perder o prazo de uma etapa, e assim notificá-lo de que precisa pedir prorrogação. Esta área está subdividida em 4 subseções:

**Etapas** – Seção onde estão cadastradas todas as etapas dos cursos do PGC (e.g., prova de inglês, exame de qualificação e pedido de banca).

**Realização de Etapas** – Seção onde são cadastradas as realizações de etapas de cada matrícula, associando-a a uma etapa em uma data de conclusão. A Figura 2.3 apresenta a tela de realização de etapas.

**Tipos de Prorrogação** – Seção onde estão cadastradas as possíveis prorrogações para cada uma das etapas. Por exemplo, a etapa pedido de banca pode ser prorrogada por três tipos diferentes de prorrogação: regular, extraordinária e final.

**Prorrogações** – Seção onde uma matrícula é relacionada a uma prorrogação, estendendo o prazo que este aluno tem para realizar uma determinada etapa.

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

3 Registros Encontrados

**Figura 2.3: Tela de realização de etapas**

## 2.1.5 FORMAÇÃO

A área de formação armazena dados básicos sobre instituições de ensino e seus cursos, afim de mitigar possíveis erros ou duplicação de dados. O cadastro de uma instituição é feito através de seu nome e sigla. Já o curso é cadastrado de acordo com seu nível e é relacionado a uma instituição. A Figura 2.4 apresenta a tela de formação.

Nome	Sigla	Cursos
Universidade Federal Fluminense	UFF	Ciência da Computação - Universidade Federal Fluminense - (Mestrado), Engenharia da Computação - Universidade Federal Fluminense - (Mestrado), Ciência da Computação - Universidade Federal Fluminense - (Doutorado)
Universidade Federal do Rio de Janeiro	UFRJ	
Universidade de São Paulo	USP	
Universidade do Estado do Rio de Janeiro	UERJ	

4 Registros Encontrados

**Figura 2.4: Tela de formação**

### 2.1.6 LOCALIDADES

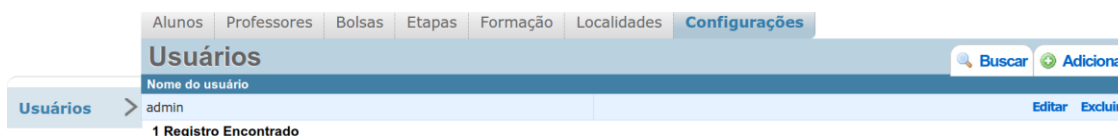
A área de localidades contém o registro de países, estados e cidades. Como a área anterior, tem como principal função mitigar possíveis erros e duplicação de dados. A Figura 2.5 apresenta a tela de estados em localidades.



**Figura 2.5: Tela de localidades (estados)**

### 2.1.7 CONFIGURAÇÕES

A área de configurações foi criada para que os administradores pudessem controlar o acesso ao sistema. Nesta área é possível fazer a inclusão e exclusão de usuários, e também alterações de seus dados como nome e senha. A Figura 2.6 apresenta a tela de configurações.



**Figura 2.6: Tela de configurações**

## 2.2 SAPOS 2

O SAPOS 2 trouxe importantes novidades ao sistema, como: melhorias no controle de etapas e prorrogações, busca avançada de matrículas, controle de credenciamento, controle de disciplinas e novos relatórios. Estes são os tópicos abordados nesta seção.

### 2.2.1 NOVO CONTROLE DE ETAPAS E PRORROGAÇÕES

Os períodos letivos da Universidade Federal Fluminense (IC-UFF) se iniciam em março e agosto. A partir desta informação, podemos notar que um semestre contempla 5 meses do ano e o outro 7, tornando o semestre uma unidade de tempo não regular.

Tendo em vista este fato, ao criar etapas, eram necessárias duas instâncias de uma etapa para cada etapa real, uma sendo a etapa no primeiro semestre, e a outra sendo a mesma etapa, porém no segundo semestre. O mesmo vale para as prorrogações. No

SAPOS 2 este sistema foi aprimorado, adicionando um sistema de controle de semestre que consegue calcular automaticamente a duração do semestre.

Além disso, também passou a ser possível a criação de etapas e prorrogações com durações que não fossem apenas de semestres, mas também de meses e dias. Por exemplo, isso permite o cadastro de uma prorrogação que conceda 3 meses a mais para que a etapa de pedido de banca seja concluída.

### 2.2.2 BUSCA AVANÇADA DE MATRÍCULAS

Um novo sistema de busca, mais robusto, foi implementado para a área de matrículas. Com esta nova busca avançada de matrículas é possível encontrar uma matrícula por várias de suas características, inclusive características de outras entidades que não sejam matrícula, como apresentado na Figura 2.7:

**Nome do Aluno** – Consulta matrículas em que o aluno possui o nome escolhido.

**Ativo** – Consulta matrículas que estejam ou não ativas, de acordo com a opção selecionada.

**Possui Bolsa** – Consulta matrículas que possuam ou não bolsa.

**Orientador** – Consulta matrículas onde o professor escolhido é o orientador.

**Realização de Etapa** – Consulta matrículas que tenham concluído a etapa selecionada em uma data igual ou inferior à data escolhida.

**Etapa Atrasada** – Consulta matrículas que estarão com a etapa selecionada atrasada em uma data igual ou posterior à data escolhida.

**Figura 2.7: Busca avançada de matrículas**

### 2.2.3 CONTROLE DE CREDENCIAMENTOS

Como foi explicado na seção 2.1.2, cada professor pode ter até 8 pontos de orientação, sendo 1 ponto quando ele for o único orientador credenciado do aluno, e meio ponto quando ele for coorientador com um outro professor credenciado no PGC. Porém, o SAPOS 1 não sabia distinguir professores que eram ou não credenciados ao PGC, e, por este motivo, fazia um cálculo errado dos pontos de orientação. Caso um professor não credenciado ao PGC seja coorientador de um aluno junto de um professor credenciado, esse último deveria somar um ponto e não meio ponto, pois o professor não credenciado não recebe nenhuma pontuação pelo fato de não ser credenciado.

Para resolver este problema, o controle de credenciamento foi criado, onde é possível criar um novo credenciamento, editar um credenciamento já existente e excluir um credenciamento. A Figura 2.8 apresenta a tela de credenciamentos.

Orientador	Nível	Editar	Excluir	Visualizar
Albus Dumbledore	Doutorado			
Minerva McGonagall	Doutorado			
Alastor Moody	Mestrado			
Severus Snape	Mestrado			
Albus Dumbledore	Mestrado			

5 Registros Encontrados

**Figura 2.8: Tela de credenciamentos**

### 2.2.4 CONTROLE DE DISCIPLINAS

O Controle de Disciplinas é uma área que foi criada para englobar o controle de diversos dados relativos às disciplinas, como área de pesquisa, tipo de disciplina, inscrição dos alunos e alocação de salas e horários. Esta área está subdividida nas seguintes seções:

**Áreas de Pesquisa** – Agrupa as disciplinas em suas respectivas áreas de pesquisa.

**Tipos de Disciplina** – Seção que faz o controle dos possíveis tipos de disciplina (e.g., obrigatória do curso e estudo orientado)

**Disciplinas** – Parte central da área de disciplinas, onde é realizado o controle, cadastro e alteração de disciplinas. A Figura 2.9 apresenta a tela de disciplinas.

**Turmas** – Seção que armazena as turmas cadastradas para cada disciplina em um semestre letivo.

**Inscrições** – Seção que faz o controle da inscrição de alunos nas turmas, armazenando sua nota, sua situação atual, e se foi aprovado ou reprovado.

**Alocações** – Seção que controla as alocações de salas e horários para cada turma. Contém restrições para que dados inválidos não sejam cadastrados, como horário de início da aula maior que o horário de fim.

Nome	Código	Área de Pesquisa	Créditos	Tipo de Disciplina
Controle dos Elementos	HOG00502	005 - Encantamento	8	Obrigatória da Área
Criaturas	HOG00301	003 - Defesa Contra Artes das Trevas	4	Obrigatória da Área
Disfarces	HOG00204	002 - Transfiguração	2	Optativa
Fórmula de Transformação	HOG00200	002 - Transfiguração	4	Obrigatória da Área
Hipnose	HOG00501	005 - Encantamento	4	Obrigatória da Área
Levitação	HOG00500	005 - Encantamento	4	Obrigatória da Área
Plantas e Seus Usos	HOG00400	004 - Herbologia	4	Obrigatória da Área
Poções	HOG00403	004 - Herbologia	6	Optativa
Proteção Contra Encantamentos	HOG00302	003 - Defesa Contra Artes das Trevas	8	Obrigatória da Área
Quadribol	HOG00102	001 - Voo com Vassoura	4	Optativa
Raízes	HOG00402	004 - Herbologia	4	Obrigatória da Área
Sementes	HOG00401	004 - Herbologia	6	Obrigatória da Área
Transformação Humana	HOG00203	002 - Transfiguração	6	Obrigatória da Área
Transformação em Animais	HOG00202	002 - Transfiguração	4	Obrigatória da Área
Transformação em Objetos	HOG00201	002 - Transfiguração	4	Obrigatória da Área

**Figura 2.9: Tela de disciplinas**

## 2.2.5 NOVOS RELATÓRIOS DO SAPOS 2

A versão inicial do SAPOS era capaz de gerar relatórios das telas de Bolsas, Alocação de Bolsas e Orientações. Na segunda versão, outros três relatórios foram criados, para facilitar o uso do sistema e também para melhorar a comunicação das informações dentro do PGC: listagem de matrículas, pauta de turmas e histórico escolar.

A listagem de matrículas consiste em um PDF com todas as matrículas cadastradas no sistema, respeitando os filtros aplicados. A pauta de turmas é preenchida pelos professores de cada turma e entregue à secretaria, para esta cadastrar os dados (notas e frequência) no sistema. Já o histórico escolar contém os dados de todas as disciplinas cursadas por um dado aluno. A pauta de turmas pode ser vista na Figura 2.10 e o histórico escolar na Figura 2.11.



Universidade Federal Fluminense  
Instituto de Computação  
Pós-Graduação



#### RESUMO SEMESTRAL

NOME DA DISCIPLINA				SEMESTRE/ANO		AULAS DADAS
Proteção Contra Encantamentos				1º/2013		
Nº	Matrícula	Nome do Aluno	Nota Final	Freq S/I	Situação	Obs.:
1	DH01	Charlie Weasley	10.0	S	Aprovado	
2	DH02	Bill Weasley			Incompleto	
3	MH01	Harry Potter			Incompleto	
4	MH03	Hermione Granger	6.0	S	Aprovado	
5	MH02	Ronald Weasley			Incompleto	
6	MH04	Draco Malfoy			Incompleto	
7	MH06	Luna Lovegood	0.0	I	Reprovado	

Severus Snape

**Figura 2.10: Pauta de turma**

Universidade Federal Fluminense  
Instituto de Computação  
Programa de Pós-Graduação em Computação



#### Histórico Escolar

Aluno:	Hermione Granger		
Matrícula:	MH03	Naturalidade:	Londres
		Data de Nascimento:	1979-09-19
Identidade:	33.333.333-3	Órgão Expedidor:	Dep. Bruxas
C.P.F.:	333.333.333-33		
Curso:	Mestrado em Computação		
Mês/Ano de ingresso no curso:	Agosto/2012		
Mês/Ano de desligamento do curso:	--	Motivo:	--

Período	Código	Nome da Disciplina	Nota	Créditos
2/2012	HOG00100	Voo I	9.5	4
2/2012	HOG00502	Controle dos Elementos	8.0	8
2/2012	HOG00403	Poções	AP	6
2/2012	HOG00503	Tópicos Avançados em Fogo	7.0	4
1/2013	HOG00302	Proteção Contra Encantamentos	6.0	8
1/2013	HOG00101	Voo II	8.0	4
1/2013	HOG00400	Plantas e Seus Usos	10.0	4
1/2013	HOG00500	Levitação	7.0	4
1/2013	HOG00402	Raizes	10.0	4
1/2013	HOG00401	Sementes	10.0	6
<b>Total</b>				52

#### Etapas obrigatórias concluídas

Agosto/2012 Exame de Qualificação  
Setembro/2012 Prova de Inglês

Niterói, 01 de Maio de 2014	<p>Este documento só é válido sem rasuras, com selo da UFF e com a assinatura do Coordenador.</p> <p>_____</p> <p>ASSINATURA DO COORDENADOR</p>	Pág. nº 1
-----------------------------	---	-----------

**Figura 2.11: Histórico escolar**

## **2.3 DISCUSSÃO**

Neste capítulo foi apresentada uma visão geral das versões anteriores do SAPOS, iniciando com o protótipo criado pelos Prof. Leonardo Murta e Vanessa Braganholo a pedido do Coordenador da Pós-Graduação em Computação, Prof. Celso C. Ribeiro, seguindo para o SAPOS 1, TCC dos alunos Tiago Amaro e Rodrigo Ferreira, e terminando com o SAPOS 2, TCC do aluno Bruno Schettino.

Diversas funcionalidades foram implementadas nas primeiras duas versões do SAPOS. Nos próximos capítulos serão abordadas as funcionalidades criadas para a terceira versão do SAPOS, objeto deste trabalho. O Apêndice A apresenta a diferença entre o modelo de classes das versões 2 e 3 do SAPOS.

## CAPÍTULO 3 – SAPOS 3: NOTIFICAÇÕES

Como apresentado anteriormente, a segunda versão do SAPOS trouxe diversas melhorias em consultas, relatórios e controle de informações referentes à Pós-Graduação. Algumas dessas consultas precisavam ser feitas de forma recorrente com uma determinada frequência. Por exemplo, todo mês era necessário verificar se o prazo de alguma etapa (exame de qualificação, pedido de banca ou prova de inglês) de algum aluno se esgotaria no mês seguinte e avisá-lo para realizar a etapa ou pedir uma prorrogação. Esta consulta pode ser feita pela tela de consulta de matrículas, mas avisar cada aluno exigia um esforço de procurar seu contato e então contatá-lo por fora do SAPOS. Com isso, surgiu a necessidade de permitir que o SAPOS notificasse alunos e professores automaticamente, de forma programada, a partir de determinadas consultas.

Além da necessidade do envio de notificações programadas, verificou-se que era desejável enviar notificações para alertar algumas alterações de estados no sistema. Por exemplo, após a secretaria lançar ou alterar notas de uma turma no sistema, por questões de segurança, era desejável enviar um e-mail para o professor com as notas lançadas, para que ele pudesse conferir, e ter a garantia de sempre saber quais foram as alterações. Este tipo de evento seguido de algum alerta motivou a criação de notificações fixas.

Este capítulo apresenta as motivações e decisões por trás da implementação do sistema de notificações do SAPOS 3. A Seção 3.1 apresenta notificações programadas, explicando como foi feita a implementação e como é feita a criação de uma nova notificação. A Seção 3.2 apresenta como foram feitas as implementações de notificações fixas. Por fim, a Seção 3.3 encerra este capítulo com uma discussão sobre o que foi apresentado.

### 3.1 NOTIFICAÇÕES PROGRAMADAS

Como apresentado na introdução, as notificações programadas surgiram a partir da necessidade de realizar consultas periodicamente e criar avisos a partir dessas consultas. Isso significa que toda notificação programada possui um período de execução e uma consulta. A realização de uma consulta resulta no envio de mensagens, caso haja resultados. Esta realização de consulta será chamada de **execução da notificação**.

A partir da necessidade de notificações programadas, surgiram algumas perguntas: Quando executar cada consulta? Como definir a consulta? Pra quem a mensagem deve ser enviada e qual deve ser o formato de cada mensagem?

Nas seguintes subseções essas perguntas são respondidas, de forma a mostrar como foi feita a implementação das notificações programadas. A subseção 3.1.1 discute quando ocorre a execução de cada consulta. A subseção 3.1.2 discute como definir cada consulta. A subseção 3.1.3 discute como decidir pra quem a mensagem deve ser enviada e qual deve ser o formato de cada mensagem. Em seguida, a subseção 3.1.4 apresenta como é feita a criação de uma nova notificação, condensando o que foi apresentado nas subseções anteriores.

Afim de facilitar o entendimento do funcionamento das notificações programadas, a Figura 3.1 apresenta a tela de cadastro da notificação para alunos de etapas vencidas ou que vencerão em um mês.

# Notificações

[Buscar](#) [Adicionar](#)

---

## Adicionar Notificação

Título:

Frequência:  Diária: todo dia; Semanal: toda segunda; Mensal: dia 01 de cada mês; Semestral: 01/03 e 01/08; Anual: 01/01

Offset da Notificação:  Data de execução. Frequência Mensal com offset 3d executará todo dia 04 do mês

Offset da Consulta:  Data de consulta. Frequência Semestral com offset -1M terá variavel para dias 01/02 ou 01/07

Consulta

Variáveis:

- %{query\_date}  
Data de consulta.  
Ex: 15/01/2014, para frequencia anual e offset de consulta 14d
- %{this\_semester\_year}  
Ano do semestre da data de consulta. Ex: 2013, para data de consulta 15/01/2014
- %{this\_semester\_number}  
Número do semestre da data de consulta. Ex: 2, para data de consulta 15/01/2014
- %{last\_semester\_year}  
Ano do semestre anterior à data de consulta. Ex: 2013, para data de consulta 15/01/2014
- %{last\_semester\_number}  
Número do semestre anterior à data de consulta. Ex: 1, para data de consulta 15/01/2014

```

1|SELECT students.email AS email,
2|    students.name AS name,
3|    phases.name AS phase_name,
4|    due date AS due date
5|FROM phase completions
6|    INNER JOIN enrollments
7|        ON enrollments.id = phase completions.enrollment id
8|    INNER JOIN students ON students.id = enrollments.student_id
9|    INNER JOIN phases ON phases.id = phase completions.phase_id
10|WHERE due date<=%{query_date}
11|AND enrollments.id NOT IN (SELECT dismissals.enrollment_id
12|                            FROM dismissals)
13|AND enrollments.enrollment status_id = 2
14|AND completion_date IS NULL
15|/* Se fase for artigo A1 só vale para alunos que entraram a partir de 2011
16|AND (phases.id <> 5 OR enrollments.admission_date >= '2011-03-01')
17|/* Se for artigo para mestrado, só vale para alunos que entraram a partir
18|AND (phases.id <> 6 OR enrollments.admission_date >= '2014-03-01')

```

Individual ☒ Um e-mail para cada resultado

Template do Destinatário:

Template do Assunto:

Template do Corpo

Os campos de template utilizam linguagem ERB. Ver detalhes

Variáveis:

- <% records %>  
Lista com todos os resultados
- <% record %>  
Mapa dos atributos de cada resultado. Disponível apenas para notificações individuais.  
Ex: <%=record['email'] %>  
\* O mesmo valor pode ser obtido por <%=email %>

Funções:

- <%=localize(var, formato)%>  
Imprime tempo com formato melhor. Opções de formato:  
:default, :defaultdate, :short, :shortdate, :long, :longdate, :picker, :day, :monthyear, :monthyear2

```

1|<%= name %>
2|
3|Informamos que não consta em nosso sistema o cumprimento da etapa <%= phase_name %>.
4|
5|Caso você já tenha realizado os procedimentos para o cumprimento dessa etapa, clique aqui para atualizar sua situação.
6|
7|Regras para Pedidos de Banca:
8|http://www.ic.uff.br/images/documentos/pos_graduacao/regras_e_procedimentos.pdf
9|
10|-----
11|
12|Regras para Exame de Qualificação:
13|http://www.ic.uff.br/images/documentos/pos_graduacao/regras_e_procedimentos.pdf
14|
15|-----
16|
17|Regras para Prova de Inglês:
18|http://www.ic.uff.br/images/documentos/pos_graduacao/regras_e_procedimentos.pdf
19|(itens 4 e 5).
20|
21|

```

**Figura 3.1: Cadastro de Notificação Programada**

Ao longo desta seção, esta figura será revisitada para apoiar a explicação do funcionamento das notificações programadas e dos seguintes campos, que podem ser vistos na figura:

**Frequência** – Define momentos base de execução da consulta. Ver Tabela 3.1.

**Deslocamento da Notificação** – Define a data de execução. Ver Tabela 3.2.

**Deslocamento da Consulta** – Define a data usada na consulta. Ver Tabela 3.2.

**Consulta** – Define o SQL de consulta, que usa variáveis definidas na Tabela 3.3.

**Individual** – Define se será enviado um e-mail para cada resultado ou se será enviado um e-mail com todos os resultados.

**Modelo do Destinatário** – Define quem é (são) o(s) destinatário(s).

**Modelo do Assunto** – Define qual é o assunto de cada mensagem.

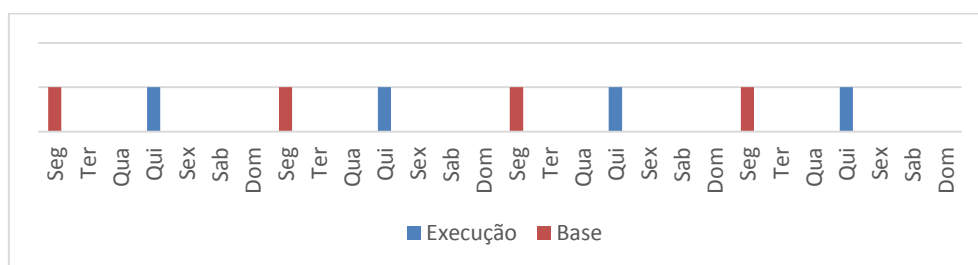
**Modelo do Corpo** – Define qual é o corpo de cada mensagem.

Os campos de Modelo utilizam a linguagem *ERB*, descrita na Tabela 3.4, podendo utilizar variáveis descritas na subseção 3.1.3, e a função *localize* com algum parâmetro da Tabela 3.5.

### 3.1.1 AGENDAMENTO DE NOTIFICAÇÕES

A consulta pode ser executada uma vez por dia, uma vez por semana, uma vez por mês, uma vez por semestre e até mesmo uma vez por ano. Contudo, apenas definir a frequência de execução não permite definir precisamente quando a notificação deve ser executada: Em que dia da semana uma notificação com frequência semanal deve ser executada?

Para responder esta pergunta foi definido um sistema de deslocamento (*offset*) em conjunto com a definição de momentos base para cada frequência, permitindo uma melhor configuração de quando cada notificação deve ser executada. Ou seja, sabendo-se que a frequência semanal ocorre toda segunda-feira (momento base), se o deslocamento da notificação for de 3 dias, esta será executada toda quinta-feira, como pode ser visto na Figura 3.2.



**Figura 3.2: Frequência ajustada por deslocamento**

A Tabela 3.1 mostra qual é o momento base para cada frequência disponível no SAPOS. Apesar da possibilidade da utilização de frequência diária e semanal, atualmente nenhuma notificação do SAPOS utiliza essas frequências. Por conta disso, nenhuma frequência com granularidade menor do que “Dia” foi implementada. Todos momentos base definem o horário como 00:00 (meia-noite).

**Tabela 3.1: Data de execução base para cada frequência**

Frequência	Base
Diária	Todo dia
Semanal	Toda segunda-feira
Mensal	Todo dia 01 do mês
Semestral	Datas: 01/03 e 01/08
Anual	Data: 01/01

Para definir o deslocamento foi adotada uma notação baseada na notação do *Rufus-Scheduler* (METTRAUX, 2014), onde se usa o número seguido, opcionalmente, da unidade. A Tabela 3.2 apresenta uma relação de unidades para o deslocamento.

**Tabela 3.2: Relação de unidades para deslocamento**

Unidade	Significado	Exemplo	Leitura do Exemplo
y	Ano	1y	Um ano
M	Mês	2M	Dois meses
w	Semana	3w	Três semanas
d	Dia	4d	Quatro dias
h	Hora	5h	Cinco horas
m	Minuto	6m	Seis minutos
s	Segundo	7s	Sete segundos
(sem unidade)	Dia	8	Oito dias

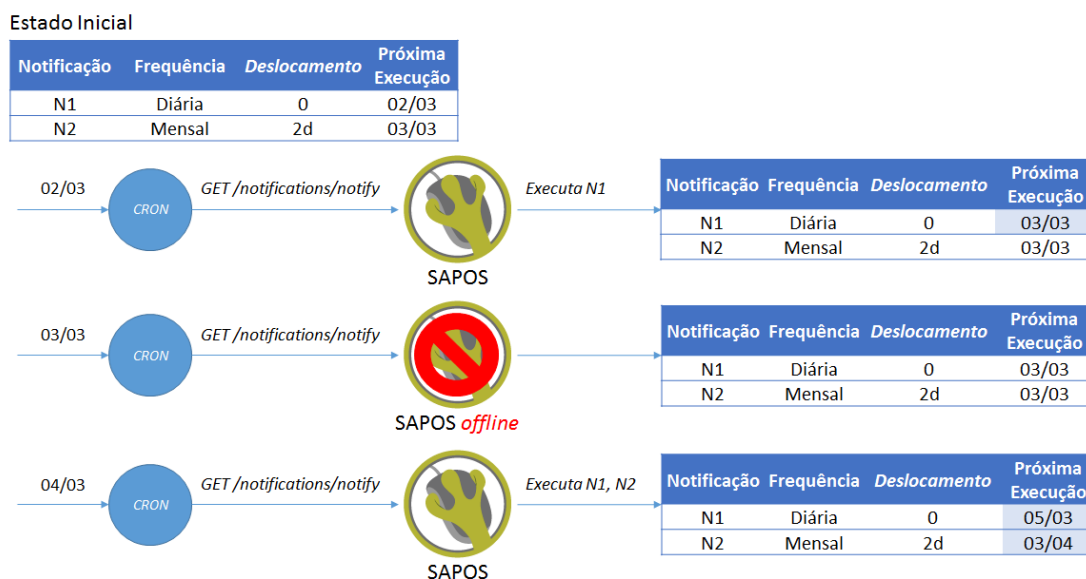
O deslocamento também pode ser uma composição de unidades: “2w1d” significa “duas semanas e um dia”. Além disso, o deslocamento também pode ser negativo. Uma notificação com frequência mensal e deslocamento “-1d” ficará agendada para o último dia de cada mês. Na Figura 3.1, a frequência foi configurada como “Mensal” e o deslocamento foi configurado como “-20d”, portanto 20 dias antes do término de cada mês, esta notificação será executada.

Nesta relação de deslocamentos, a unidade de mês é a única que possui um deslocamento variável. Um mês pode significar 28, 29, 30 ou 31 dias, dependendo do mês em que se está. No entanto, a data 01/02 + “1M” sempre resultará na data 01/03, independente de fevereiro possuir 28 ou 29 dias (ano bissexto).

Esta forma de agendamento permite agendar a execução de notificações para datas precisas de acordo com cada necessidade. Entretanto, a definição de datas precisas para execução pode ocasionar problemas se o servidor estiver desligado no momento desejado. O servidor pode estar desligado no momento da execução por diversos motivos, tais como queda de luz ou implantação (*deploy*) de uma nova versão exatamente no momento de execução da notificação.

Para resolver este problema, foi feita uma separação do agendamento entre a data esperada para a próxima execução e um sistema de chamada (*pulling*) com frequência fixa. O sistema de *chamada* verifica em um dado momento quais são todas as notificações cuja data esperada para próxima execução é anterior a este e as executa. Após a execução da notificação, uma nova data é calculada para próxima execução e armazenada. Desta forma, se a chamada for executada com frequência de um dia às 04:00, uma notificação estiver marcada para executar meia-noite da segunda-feira e o servidor estiver desligado no momento de execução da *chamada* na segunda-feira, no dia seguinte (terça-feira) será verificado, durante a execução da *chamada*, que a notificação ainda não foi executada.

Este mecanismo foi implementado a partir da criação de uma *URL* para executar a operação de chamada (*/notifications/notify*) com a definição de uma tarefa *cron* (COGNITION, 1999) que realiza uma operação de *GET* nesta *URL* a partir de uma frequência agendada pelo *cron*. Para o contexto do SAPOS, foi utilizado o “*cron.daily*”, visto que a execução diária trata qualquer uma das frequências disponíveis. A Figura 3.3 ajuda a entender como ocorre este mecanismo.



**Figura 3.3: Agendamento de Notificações Programadas**

Esta figura mostra o agendamento de duas notificações N1 e N2, sendo N1 diária e N2 mensal com deslocamento de dois dias. Inicialmente N1 tem a próxima execução marcada para o dia 02/03, enquanto N2 tem a próxima execução marcada para o dia 03/03. No dia 02/03, o *cron* é executado e realiza um *GET* na *URL* de *pulling* do SAPOS. O sistema verifica que no dia 02/03 deve executar a notificação N1. Em seguida, executa-a e calcula a data da próxima execução para o dia 03/03. No dia 03/03, o *cron* tenta realizar



o *GET*, mas o servidor está *off-line*, portanto nenhuma notificação é executada, apesar das datas marcadas para o dia 03/03. Já no dia 04/03, o *cron* realiza o *GET* e o SAPOS percebe que as duas notificações devem ser executadas, por não terem sido executadas no dia anterior. Após a execução de N1 e N2, as datas são atualizadas respectivamente para 05/03 e 03/04.


É importante notar nesse esquema que a execução de notificações não é retroativa. Como a notificação N1 é diária, teoricamente deveriam ocorrer três execuções em três dias, mas apenas duas ocorreram e uma delas nunca será compensada.

### 3.1.2 REALIZAÇÃO DE CONSULTAS

Para a realização de consultas, foi decidida a utilização de código SQL. Essa decisão surgiu da necessidade da coordenação do PGC em executar consultas complexas em SQL que não são suportadas pelo SAPOS. Atualmente, as consultas comuns suportadas no SAPOS são diretamente relacionadas a consultas do *Active Scaffold* (WHITE, 2013), que permitem apenas retornar uma entidade específica, sem a possibilidade de alterar a visualização do resultado. Ou seja, se a matrícula mostra normalmente os campos “Número de Matrícula”, “Nome do Aluno”, “Tipo de Matrícula”, “Nível”, “Data de Admissão” e “Desligamento”, uma consulta em matrículas pelo SAPOS vai mostrar esses mesmos campos, como pode ser visto na Figura 3.4, sem a possibilidade de escolher outros campos, nem mesmo de utilizar funções do SQL como *SUM*, *COUNT* e outras.

A coordenação precisava realizar consultas mais complexas, como a consulta do número de alunos ativos em uma determinada data, que pode ser vista na Figura 3.5. Esta consulta retorna *COUNT* agrupado por *level\_id* e não ocorre em uma única entidade, mas na junção de matrículas com níveis. Por conta disso, foi percebido que as consultas do *Active Scaffold* atualmente implementadas no SAPOS, não são completas o suficiente para o envio de notificações.

Versão 3.3.0-9-gd8e3ff5 | Créditos



# SAPOS

Sistema de Apoio à Pós-Graduação

Desenvolvido pelo Instituto de Computação - UFF

Alunos

Desligamentos

**Matrículas**

Níveis

Razões de Desligamento

Tipos de Matrícula

Alunos | Professores | Bolsas | Etapas | Disciplinas | Formação | Localidades | Configurações | Logout

## Matrículas

Número de Matrícula:

Nome do Aluno:

Nível:

Tipo de Matrícula:

Data de Admissão:

Ativo?:

Possui bolsa?:

Orientador:

Realização de Etapa:  até

Etapa atrasada:  em

Disciplina:  em

Número de Matrícula	Nome do Aluno	Tipo de Matrícula	Nível	Data de Admissão	Desligamento	
DH01	Bill Weasley	Regular	Doutorado	Março-2013		<input type="button" value="Editar"/> <input type="button" value="Excluir"/> <input type="button" value="Ver"/>
MH02	Harry Potter	Regular	Mestrado	Março-2013		<input type="button" value="Editar"/> <input type="button" value="Excluir"/> <input type="button" value="Ver"/>
MH05	Ronald Weasley	Avulso	Mestrado	Março-2013		<input type="button" value="Editar"/> <input type="button" value="Excluir"/> <input type="button" value="Ver"/>

**Figura 3.4: Consulta de matrículas ativas**

```

SELECT count(*) AS num_alunos, levels.name AS level
FROM enrollments, levels
WHERE enrollments.id NOT IN (SELECT enrollment_id
                             FROM dismissals
                             WHERE date < %{query_date})
AND admission_date < %{query_date}
AND enrollment_status_id = 2
AND enrollments.level_id = levels.id
GROUP BY level_id

```

**Figura 3.5: Consulta do número de alunos ativos**

Ao invés de SQL, uma opção para escrita dessa consulta poderia ter sido a utilização do Mapeamento Objeto-Relacional do *Ruby on Rails: Active Record* (HANSSON, 2014). Entretanto, esta opção exigiria o conhecimento do *Active Record* por quem fosse cadastrar as notificações programadas e exigiria modificação do código, limitando o cadastro apenas a desenvolvedores do SAPOS e não atendendo a necessidade da coordenação do PGC em executar consultas complexas. Sendo assim, a escrita de consultas em SQL foi considerada a melhor opção, por não exigir o conhecimento do *Active Record* e por permitir a escrita de consultas por um usuário especialista pela própria interface do sistema.

O SQL da consulta é executado sem nenhuma restrição, o que pode ser potencialmente perigoso para a integridade do sistema. Não há nada que impeça a execução de comandos perigosos como *DELETE* ou *DROP DATABASE* durante a criação

das notificações. Como nem sempre o coordenador entende de SQL, a criação de notificações programadas foi restringida apenas a usuários administradores do sistema, assumindo que estes possuem conhecimento técnico de banco de dados, para evitar este tipo de problemas.

Um ponto a se considerar nas notificações programadas é que muitas das consultas dependem de datas específicas. Por exemplo, na consulta que retorna o número de alunos ativos, apresentada na Figura 3.5, é necessário saber quais são as matrículas ativas na data marcada, visto que um desligamento pode ser marcado para o futuro. Se a consulta está agendada para o final de cada mês, uma matrícula possui desligamento cadastrado para junho e, por algum motivo, a consulta de maio ocorre no início de junho; o resultado de alunos ativos em maio deve ser referente à data prevista de execução (final de maio) e não à data real de execução (início de junho). Por conta disso, foram introduzidas variáveis na consulta referentes à data agendada.

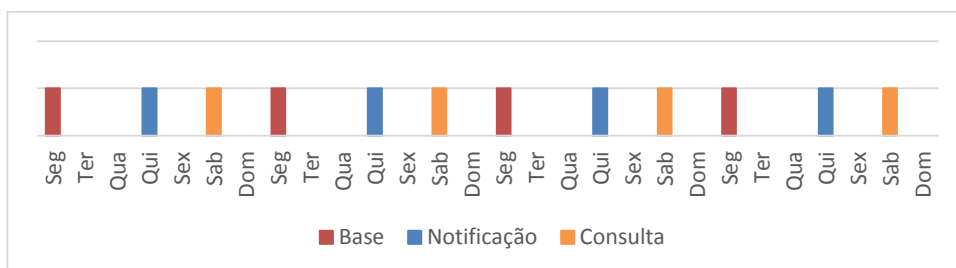
Contudo, nem sempre a data de execução da consulta é a mesma data que se deseja consultar. Para dar flexibilidade à consulta e utilizar uma data diferente da data agendada, sem a necessidade de manipular datas em SQL, foi criado um novo deslocamento no agendamento para indicar qual data deve ser usada na consulta. Sendo assim, o agendamento passou a ter:

**Frequência** – apresentada na subseção 3.1.1. Define momentos base de execução.

**Deslocamento da Notificação** – apresentada na subseção 3.1.1. Define o momento esperado para próxima execução da notificação.

**Deslocamento da Consulta** – o novo deslocamento. Define a data que será usada na consulta.

O cálculo da data de consulta segue o mesmo cálculo definido na subseção 3.1.1, utilizando momento base da frequência + deslocamento da consulta. A Figura 3.6 ilustra o uso de dois deslocamento com frequência semanal. O deslocamento de notificação marcado como “3d” e o de consulta marcado como “5d”.



**Figura 3.6: Frequência ajustada por dois deslocamentos**

Na notificação de vencimento de etapas em um mês, apresentada na Figura 3.1, enquanto a notificação é executada 20 dias antes do início do mês, a data de consulta é o primeiro dia do mês (deslocamento de consulta “0”), pois deseja-se buscar as etapas que estarão vencidas no início do mês seguinte a execução da notificação.

A utilização de variáveis no SQL é feita através da função de formatação de *String* do *Ruby* (MATSUMOTO, 2010a). Isso significa que variáveis podem ser usadas como “%{nome\_da\_variavel}” (sem as aspas). Na Figura 3.5, é possível ver a utilização da variável `query_date`. Essa variável representa a data da consulta. O uso da função de formatação transforma o caractere ‘%’ em um caractere especial, portanto para utilizar o caractere-curinga ‘%’ do SQL em um operador *LIKE*, passa a ser necessário escapá-lo com outro ‘%’. Assim, uma consulta com *LIKE* deve ficar: “campo *LIKE* “%%VALOR%%””.

Algumas consultas complexas realizadas pela coordenação utilizavam o número do ano e do semestre, portanto, algumas variáveis foram derivadas da data de consulta. A Tabela 3.3 apresenta a relação de todas as variáveis possíveis na consulta. Perceba que o semestre/ano para a data 15/01/2014 é 2/2013, visto que o semestre 1/2014 começa apenas no dia 01/03.

**Tabela 3.3: Variáveis de consulta**

Variável	Significado	Exemplo
<b>query_date</b>	Data de consulta	<b>15/01/2014</b> , para frequência anual e deslocamento de consulta 14d
<b>this_semester_year</b>	Ano do semestre da data de consulta	<b>2013</b> , para data de consulta 15/01/2014
<b>this_semester_number</b>	Número do semestre da data de consulta	<b>2</b> , para data de consulta 15/01/2014
<b>last_semester_year</b>	Ano do semestre anterior à data de consulta	<b>2013</b> , para data de consulta 15/01/2014
<b>last_semester_number</b>	Número do semestre anterior à data de consulta	<b>1</b> , para data de consulta 15/01/2014

### 3.1.3 ENVIO DE MENSAGENS

Toda consulta que retorna algum resultado gera o envio de uma ou mais mensagens. Quando a consulta retorna um resultado vazio, nenhuma mensagem é enviada. As mensagens são enviadas por e-mail. O servidor de e-mails é configurado nos arquivos de ambiente pela variável *action\_mailer* do *Ruby on Rails*. Para o servidor de produção do SAPOS no PGC, é usado o *sendmail* (COSTALES et al., 2007) do *Unix*.

Um e-mail é composto de **destinatário**, **assunto** e **corpo** e é importante que cada tipo de notificação tenha um modelo (*template*) diferenciado. Por exemplo, em notificações de vencimento de etapas, é necessário mostrar, no corpo do e-mail, quais são as regras pra cada etapa. Para definir o modelo de cada um desses campos, foi usada a linguagem de modelos *ERB* (MATSUMOTO, 2010b).

*ERB* significa “Embedded Ruby” (*Ruby* embarcado). Esta linguagem permite executar código *Ruby*, com variáveis, funções, condições e iterações dentro de um modelo. Existem quatro principais usos de *tags* na linguagem, como pode ser visto na Tabela 3.4.

**Tabela 3.4: Tags do ERB**

Tag	Significado	Exemplo	Resultado
<% %>	Executa código <i>Ruby</i>	<% if true %> a <% else %> b <% end %>	a
<%# %>	Comentário	<%# Comentário %> b	b
<%= %>	Imprime o output do código <i>Ruby</i>	<%= [1, 2, 3].join(', ') %>	1, 2, 3
<%- ou -%>	Remove espaço em branco antes ou depois	<%= "c" -%> d	cd

Em uma notificação programada, pode-se desejar que o SAPOS envie apenas um e-mail de relatório com todos os resultados da consulta, como é o caso da consulta pelo número de alunos ativos, em que a coordenação deseja receber em um único e-mail a quantidade de alunos ativos de acordo com o nível (mestrado ou doutorado); ou pode-se desejar que o SAPOS envie um e-mail para cada tupla retornada, como é o caso da consulta de vencimento de etapas, apresentada na Figura 3.1, em que cada aluno deve ser notificado apenas do vencimento de suas etapas. Por conta disso, foi adicionada a opção

**Individual**, no cadastro da notificação, que indica se um e-mail será enviado individualmente para cada tupla, ou se será um único e-mail para todos resultados.

A coordenação pode ser composta por mais de uma pessoa e é interessante que todos recebam os e-mails “únicos” de relatório. Portanto, quando se quer que mais de uma pessoa receba o mesmo e-mail, pode-se separar os destinatários de uma mesma notificação por “;” (ponto e vírgula). Ou seja, o campo de destinatário pode ser “joaofelipenp@gmail.com; lvieira@lvieira.com” ou até mesmo usar *ERB* para definir destinatários que dependam do resultado da consulta como “<%= email %>” ou consultar destinatários usando *Active Record* para obter a lista de usuários cadastrados como coordenador no sistema: “<%= user.where(:role\_id => 2).map(&:email).join(';') %>”.

As variáveis disponíveis para uso no modelo dependem da consulta SQL e da opção Individual. Se for enviado um único e-mail com todos os resultados, a única variável disponível é a “records”, que é uma lista com todos os resultados obtidos. A Figura 3.7 apresenta um modelo que lista todos os coordenadores. Como pode ser visto na figura, ocorre uma iteração em “records” e os campos são acessados de acordo com o nome das colunas no *SELECT*.

```

Consulta:
SELECT email, name AS user_name
FROM users
WHERE role_id = 2

(E-mail único)

Template:
<% records.each do |record| -%>
- <%= record['user_name'] %> (<%= record['email'] %>)
<% end %>

```

**Figura 3.7: Exemplo de modelo para e-mail único**

Já para o caso em que é enviado um e-mail para cada tupla, existe uma variável para cada nome de coluna e existe a variável “record” para o resultado. A variável “records” também existe nessa situação, mas, em geral, não há muita razão em usá-la. A Figura 3.8 apresenta um modelo que mostra informações de um coordenador e diz o total de coordenadores no sapos.

**Consulta:**  
**SELECT** email, name **AS** user\_name, created\_at  
**FROM** users  
**WHERE** role\_id = 2

**(E-mail para cada tupla)**

**Template:**  
 Coordenador:  
 - <%= user\_name %> (<%= record['email'] %>)  
 - Criado em: <%= localize(created\_at, :defaultdate) %>

Número de coordenadores:  
 <%= records.size %>

**Figura 3.8: Exemplo de modelo para e-mail para cada tupla**

Perceba que <%= user\_name %> tem exatamente o mesmo resultado que <%= record['user\_name'] %> e não há nenhuma regra que indique qual deve ser usado em cada momento, deixando a escolha para quem for criar a notificação.

Além de variáveis, o *ERB* também permite utilizar funções. Na Figura 3.8, é possível observar a função *localize*. Essa função recebe dois parâmetros: uma data/hora e um formato. Este formato indicará como a data deve ser mostrada na mensagem. A Tabela 3.5 apresenta os possíveis formatos para a data/hora 26/02/2014 23:10:05.

**Tabela 3.5: Formatos para localize**

Formato	<i>strftime</i> (MATSUMOTO, 2010c)	Exemplo
:default	%A, %d de %B de %Y, %H:%M h	Quarta-feira, 26 de Fevereiro de 2014, 23:10 h
:defaultdate	%d-%m-%Y	26/02/2014
:short	%H:%M h	23:10 h
:shortdate	%d de %B	26 de Fevereiro
:long	%A, %d de %B de %Y, %H:%M h	Quarta-feira, 26 de Fevereiro de 2014, 23:10 h
:longdate	%d de %B de %Y	26 de Fevereiro de 2014
:picker	%a, %d %b %Y %H:%M:%S	Qua, 26 Feb 2014 23:10:05
:day	%A, %d de %B de %Y	Quarta-feira, 26 de Fevereiro de 2014
:monthyear	%B-%Y	Fevereiro-2014
:monthyear2	%B/%Y	Fevereiro/2014

O corpo da mensagem de uma notificação pode ser dividido em duas partes: modelo definido pela notificação e um rodapé padrão que é igual em todas as notificações. O rodapé pode ser definido pela variável customizável *notification\_footer*. No Capítulo 4, variáveis customizáveis serão explicadas.

Uma outra variável que será explicada no Capítulo 4 é a *redirect\_email*, que permite redirecionar todos os e-mails de notificação para um destinatário específico, ou até mesmo cancelar o envio de mensagens.

Nos arquivos de ambiente do *Rails*, existe também a variável booleana *should\_send\_emails*. Esta variável indica se notificações fixas ou programadas devem ser enviadas e pode desativar o envio de e-mails sem a presença da variável customizável *redirect\_email*. Esta variável é usada com o objetivo de desativar o envio de e-mails durante a execução de testes e durante o uso do *rails console* em desenvolvimento.

Cada mensagem enviada gera um registro de envio na tabela *notification\_logs* que armazena qual foi a notificação programada geradora, o destinatário, o assunto, o corpo da mensagem e a data de execução da notificação. A data de execução da notificação nem sempre é a mesma data do envio de e-mail, pois o servidor de e-mail pode não estar conectado à internet e enviar posteriormente, quando for conectado.

### 3.1.4 CADASTRO DE NOTIFICAÇÕES PROGRAMADAS

As notificações programadas ficam salvas no banco de dados na tabela *notifications*. Para criar uma notificação programada, é escolhido um título (opcional) e todos os campos mencionados nas subseções 3.1.1, 3.1.2 e 3.1.3 devem ser preenchidos.

A tela de criação e edição de notificações programadas foi apresentada na Figura 3.1. Nessa figura, é possível perceber que os campos “Consulta” e “Modelo do Corpo” possuem um campo de texto com realce de sintaxe. Essa coloração foi feita com o *CodeMirror* (HAVERBEKE, 2013), utilizando os modos de coloração “text/x-mysql” e “text/html”, respectivamente.

Ao clicar em “Salvar”, ocorre uma simulação da notificação em que a consulta SQL é executada e os campos de valores são calculados. Esta simulação é capaz de pegar erros na formulação do SQL ou dos modelos e evitar que os administradores sejam surpreendidos pelo erro durante o momento de execução programado. Durante esta simulação, nenhuma mensagem é enviada.

Após a criação da notificação, é possível fazer uma outra simulação a partir de um *link* na página de notificações. Esta simulação apresenta na tela quais são as mensagens que seriam enviadas para uma determinada data de consulta, como pode ser visto na Figura 3.9. Por padrão, a data de consulta é a data calculada através do *offset* para a próxima execução, mas essa data pode ser alterada através de outro *link*.



Usuários	Alunos	Professores	Bolsas	Etapas	Disciplinas	Formação	Localidades	Configurações	Logout						
Papéis	Notificações								<div>Buscar</div> <div>Adicionar</div>						
Log															
Notificações															
Notificações Enviadas															
Variáveis															
<div>Notificar Pedido de Banca a Coordenador</div> <div>Data de Consulta: 02-07-2014</div> <table><thead><tr><th>Para</th><th>Assunto</th><th>Corpo</th></tr></thead><tbody><tr><td>coordenacao@sapos.com</td><td>Alunos precisam realizar Pedido de Banca em um mês</td><td>Coordenador, Os seguintes alunos precisam pedir banca em 1 mês: - Ronald Weasley (ronald.weasley@hogwarts.com) - Fevereiro/2014</td></tr></tbody></table> <div>Fechar</div> <div>Alunos ainda não desligados</div> <div>Validar</div>										Para	Assunto	Corpo	coordenacao@sapos.com	Alunos precisam realizar Pedido de Banca em um mês	Coordenador, Os seguintes alunos precisam pedir banca em 1 mês: - Ronald Weasley (ronald.weasley@hogwarts.com) - Fevereiro/2014
Para	Assunto	Corpo													
coordenacao@sapos.com	Alunos precisam realizar Pedido de Banca em um mês	Coordenador, Os seguintes alunos precisam pedir banca em 1 mês: - Ronald Weasley (ronald.weasley@hogwarts.com) - Fevereiro/2014													

**Figura 3.9: Página de Notificações**

Essa alteração da data de consulta também é válida para a operação “Enviar Agora”, um outro *link* na página de notificações, que permite executar a notificação no momento do clique, ignorando a data agendada para a próxima execução. A figura Figura 3.10 apresenta respectivamente os *links* de “Escolher Data de Consulta”, “Enviar Agora” e “Simular”.



**Figura 3.10: Links da Página de Notificações**

No Anexo A encontram-se as notificações programadas atualmente cadastradas no servidor em produção do PGC da UFF.

### 3.2 NOTIFICAÇÕES FIXAS

As notificações fixas surgiram da necessidade de alertar mudanças específicas no sistema a alunos, professores e orientadores. Como os contextos de envio de notificações fixas eram muito específicos, elas foram implementadas diretamente no código, diferentemente das notificações programadas que possuem uma tabela no banco de dados para serem armazenadas. As notificações fixas são enviadas após determinados eventos. Por exemplo, após salvar uma inscrição em turma, é verificado se houve alteração na nota e, se tiver havido, são enviadas notificações para o aluno e seus orientadores.

Parte do sistema de envio de notificações fixas é igual ao sistema de notificações programadas:

- A configuração *should\_send\_emails* nos arquivos de ambiente do *Ruby on Rails* também pode impedir o envio de notificações fixas.
- A variável *notification\_footer* também adiciona um rodapé em toda notificação fixa.

- A variável *redirect\_email*, que será explicada no Capítulo 4, também redireciona e-mails de notificações fixas para os destinatários armazenados nela e também pode cancelar o envio de e-mails de notificações fixas.
- Após o envio de uma mensagem, esta mensagem é salva no registro de notificações enviadas.

Neste último ponto, há uma diferença entre notificações fixas e programadas. Enquanto o registro possui referência para notificações programadas, indicando qual foi a notificação geradora da mensagem, o registro não diz qual foi a notificação fixa geradora.

Os modelos de notificações fixas são definidos nos arquivos de internacionalização do SAPOS. A seguir, estão listadas todas as notificações fixas implementadas no sistema e qual é o evento que ativa cada uma.

- 1- Notificar aluno e seus orientadores sobre lançamento ou mudança de nota ou situação (aprovado / reprovado) da inscrição em uma turma. Evento: salvar inscrição de turma.
- 2- Notificar professor de turma sobre lançamento ou mudança de nota ou situação de inscrição em uma turma. Eventos: salvar inscrição de turma pela página de inscrições ou salvar turma com alteração nas inscrições.
- 3- Notificar orientador que um novo aluno foi cadastrado como orientando dele. Evento: criar orientação.
- 4- Notificar aluno e seus orientadores que ele cumpriu uma etapa. Evento: criar realização de etapa.
- 5- Notificar aluno e seus orientadores que ele conseguiu uma prorrogação de etapa. Evento: criar prorrogação.

### 3.3 DISCUSSÃO

Neste capítulo, vimos que os sistemas de notificações foram implementados com o intuito de poder alertar aos alunos, coordenadores e professores, sobre eventos que estão pra ocorrer, medidas que devem ser tomadas ou até mesmo por questões de segurança, como notificar alterações de notas.

Mostramos também a diferença de notificações programadas para fixas. Para as programadas foi fornecida uma infraestrutura que permite a criação de notificações pelos administradores, a partir da definição do momento de execução, consulta e modelos de

mensagens. Já as fixas foram implementadas diretamente no código associadas a eventos específicos.

O próximo capítulo discute quais foram as outras melhorias não relacionadas a notificações que foram adicionadas na terceira versão do SAPOS.

## CAPÍTULO 4 – SAPOS 3: OUTRAS MELHORIAS

Além das notificações, outras melhorias foram adicionadas ao SAPOS em sua terceira versão. Este capítulo explica essas melhorias. A Seção 4.1 apresenta o sistema de registro de mudanças (*logging*). A Seção 4.2 apresenta o sistema de variáveis que podem ser configuradas durante a execução. A Seção 4.3 apresenta as mudanças realizadas na geração de relatórios e o quadro de horários que foi introduzido ao sistema. A Seção 4.4 apresenta as diversas melhorias na visualização. Por fim, a Seção 4.5 conclui este capítulo apresentando outras melhorias e discutindo o que foi apresentado.

### 4.1 SISTEMA DE REGISTRO DE MUDANÇAS

Uma necessidade do PGC era armazenar quais foram as alterações feitas no sistema para verificar se houve algum erro e ocasionalmente revertê-lo, garantindo uma melhor confiabilidade na edição dos dados. Após uma busca de possibilidades de bibliotecas que adicionassem de forma simples essa funcionalidade, foi decidido o uso da biblioteca *paper\_trail* (STEWART, 2013).

A *paper\_trail* cria uma tabela “versions” no banco de dados, que armazena qual foi a entidade alterada, qual foi o id alterado, qual foi a ação (criação, remoção ou atualização), quais são os dados da versão antiga e qual foi o usuário que realizou a mudança. Esta biblioteca foi escolhida pela facilidade de definir quais entidades (KRASNER; POPE, 1988) deveriam ser versionadas, sendo necessário adicionar apenas uma linha de código na entidade (*model*) para que a biblioteca crie todos os gatilhos (*triggers*) necessários.

Para visualizar as versões, foi criada uma página com uma tabela que mostra todos os campos referentes à versão e fornece um *link* para uma visualização que permite ver os dados da versão antiga. A Figura 4.1 apresenta alguns registros de mudanças.

Como pode ser visto na figura, existem mudanças em que há link para o usuário que realizou a mudança e também há uma alteração na entidade Usuário em que não há usuário responsável. Quando não há usuário responsável, a mudança foi feita pelo servidor sem a ação de nenhum usuário. Nesse caso, essa mudança foi provocada pela ação de acessar o sistema (*login*), que armazena no banco de dados qual foi a última data que o usuário acessou o sistema.

Log					Buscar
Modelo	Versão Atual	Ação	Usuário	Modificado em	
Variável	País padrão de emissão da identidade	Atualização	João Felipe	01/04/2014 às 16:40	
<b>Atualização de Variável</b> Modelo: <b>Variável</b> Versão Atual: <b>País padrão de emissão da identidade</b> Ação: <b>Atualização</b> Usuário: <b>João Felipe</b> Versão Antiga: <b>name: País padrão de emissão da identidade</b> <b>variable: identity_issuing_country</b> <b>value: Brasil</b> <b>id: 4</b> <b>Fechar</b>					
Tipo de Disciplina	Tópicos Especiais	Atualização	Vanessa Braganholo	31/03/2014 às 23:25	
Usuário	Vanessa Braganholo	Atualização		31/03/2014 às 23:24	

**Figura 4.1: Tela de Registro de Mudanças**

A Figura 4.1 apresenta também o registro de atualização da variável *identity\_issuing\_country*. Esta variável é explicada nas Seções 4.2 e 4.4.

## 4.2 VARIÁVEIS CUSTOMIZÁVEIS

Outra melhoria da terceira versão do SAPOS foi a criação de um sistema de variáveis customizáveis que permitisse editar variáveis em tempo de execução. Esta seção apresenta de onde surgiu a necessidade dessas variáveis, como elas foram implementadas e qual é o significado de cada uma delas.

Existiam dois principais requisitos que motivaram a implementação dessas variáveis: customizar o SAPOS para outros programas de pós-graduação e alterar determinadas configurações de tempos em tempos sem reiniciar o servidor. Anteriormente, para alterar valores como o nível do conceito CAPES do programa de pós-graduação, era necessário editar no código-fonte o novo nível e reiniciar o servidor para fazer a implantação de uma nova versão.

Além da inconveniência de ter que reiniciar o servidor para alteração de um simples valor, como o conceito CAPES é uma valor que muda de instituição para instituição, essa abordagem também não era ideal para o uso do SAPOS em outros programas de pós-graduação, visto que exigia a alteração de uma linha de código que poderia dificultar a aplicação de atualizações.

Para resolver esses problemas, foi decidida a criação de uma tabela de variáveis no banco de dados chamada *custom\_variables*, com o objetivo de armazenar toda variável dependente de instituição ou que pudesse ser alterada durante a execução por algum administrador do sistema. Esta tabela possui 3 campos: nome (atual “variável”), descrição (atual “nome”) e valor. O campo **nome** é uma cadeia de caracteres (*string*), com limite de 255 caracteres, e representa o nome da variável usada no código. O campo **descrição**

também é uma cadeia de caracteres e apenas armazena uma descrição da variável que permite que o administrador saiba qual é o significado da variável. Por fim, o campo **valor** é do tipo texto (*text*), com limite de 65536 caracteres, e define o valor usado.

**Tabela 4.1: Variáveis implementadas no sistema**

Nome	Descrição	Valores Possíveis	Padrão
<i>program_level</i>	Nível do Programa na CAPES.	Qualquer valor. Idealmente número entre 1 e 7.	Nulo
<i>single_advisor_points</i>	Pontos para orientador único. Esta variável é usada para calcular pontos de orientação quando o professor é o único orientador de um aluno.	Qualquer número real.	1.0
<i>multiple_advisor_points</i>	Pontos para orientador múltiplo. Esta variável é usada para calcular pontos de orientação quando o professor divide orientação de aluno.	Qualquer número real.	0.5
<i>notification_footer</i>	Rodapé que aparece em todas as notificações.	Texto	""
<i>redirect_email</i>	E-mail para o qual todas as notificações serão redirecionadas.	Variável inexistente -> sem redirecionamento. "" -> nenhum e-mail é enviado. "e-mails" -> notificações redirecionadas para "e-mails"	N/A
<i>class_schedule_text</i>	Texto que aparece no rodapé do quadro de horários.	Texto	""
<i>identity_issuing_country</i>	País usado para busca dos estados emissores da identidade.	Nome de País ou "". Quando "", o estado emissor passa a usar campo texto por padrão.	"Brasil"

Apesar do tipo do valor ser texto, os valores são convertidos no código para o tipo correto durante o uso (e.g., ponto flutuante para *single\_advisor\_points*). Entretanto, atualmente não existe nenhuma validação para verificar se o tipo usado está correto.

Na entidade associada a tabela *custom\_variables*, existem métodos estáticos para cada uma das variáveis possíveis. Esses métodos retornam um valor padrão caso a variável não exista ou, caso contrário, o valor da variável convertido para o tipo desejado. A Tabela 4.1 apresenta todas as variáveis implementadas no sistema, com uma descrição, significado de cada valor e valor padrão.

Como pode ser visto nessa tabela, a variável *redirect\_email* possui efeitos diferentes quando a variável é inexistente, quando o valor dela está salvo como cadeia de caracteres vazia e quando o valor dela está salvo como um determinado e-mail ou conjunto de e-mails, separados por ponto e vírgula. Para que a variável seja considerada como inexistente, é necessário que não exista nenhuma tupla na tabela de variáveis cuja variável seja “*redirect\_email*”.

A variável *identity\_issuing\_country* é usada para indicar de qual país a lista de estado emissor da identidade deve ser lida. Uma explicação melhor dela é dada na Seção 4.4, que fala das melhorias na visualização do sistema.

As variáveis *single\_advisor\_points* e *multiple\_advisor\_points* definem o peso dos pontos de orientação. Uma melhor explicação do que são pontos de orientação e como são calculados no Programa de Pós-Graduação da UFF foi dada por SCHETTINO (2013).

### 4.3 GERAÇÃO DE RELATÓRIOS

Na seção anterior, o sistema de variáveis foi apresentado. Uma das variáveis introduzidas foi *class\_schedule\_text*, que define o texto que aparece no rodapé do quadro de horários. O quadro de horários é um relatório adicionado na terceira versão do SAPOS. Além da adição desse relatório, diversos outros relatórios foram modificados. Esta seção apresenta a geração de quadro de horários e, em seguida, descreve quais foram as alterações nos outros relatórios.

Para gerar o quadro de horários de um determinado período no SAPOS 3, deve-se buscar por ano/semestre na tela de buscas de turmas e clicar em “Quadro de Horários”. O quadro de horários gerado apresenta uma listagem de disciplinas, horários, salas e professores. A Figura 4.2 apresenta uma página de um quadro de horários.

Disciplina	Segunda	Terça	Quarta	Quinta	Sexta	Professor
Controle dos Elementos	9-11 387		9-11 387			Alastor Moody
Criaturas	*	*	*	*	*	Albus Dumbledore
Disfarces	14-16 101		14-16 101			Minerva McGonagall
Fórmula de Transformação		7-9		7-9		Remus Lupin
Hipnose		9-11		9-11		Severus Snape
Levitação		11-13 4302		11-13 4302		Alastor Moody
Plantas e Seus Usos		14-16		14-16		Albus Dumbledore
Proteção Contra Encantamentos			9-11		9-11	Albus Dumbledore
Quadribol			11-13		11-13	Minerva McGonagall
Transformação em Animais	*	*	*	*	*	Albus Dumbledore
Transformação em Objetos	14-16		14-16			Minerva McGonagall
Técnicas de Cura		7-9	7-9			Remus Lupin
Tópicos Avançados em Fogo (A1)		9-11	9-11			Severus Snape
Voo I	7-9 Pátio 1		7-9 Pátio 3.5			Remus Lupin
Voo II		14-16			14-16	Alastor Moody

\* Horário a combinar

\*\* Alunos interessados em cursar disciplinas de Tópicos Avançados devem consultar os respectivos professores antes da matrícula.

**Figura 4.2: Relatório de quadro de horários**

Algumas disciplinas devem mostrar o nome da turma junto do nome da disciplina, como ocorre com “Tópicos Avançados em Fogo”, que mostra a turma “A1” nesse quadro de horários. Para identificar quais disciplinas devem mostrar o nome da turma no quadro de horários, foi criado um atributo booleano em Tipos de Disciplina. No PGC, apenas as disciplinas de tópicos especiais devem mostrar o nome da turma. Além disso, algumas disciplinas, como dissertação e tese não devem aparecer no quadro de horários. Por conta disso, um outro atributo booleano foi adicionado em Tipos de Disciplina para indicar que tipos de disciplina não devem aparecer no quadro de horários.

A mensagem “Horário a combinar” presente no final de cada página do quadro de horários só aparece quando alguma disciplina não tem horário marcado. A mensagem “Alunos interessados em cursar disciplinas de Tópicos Avançados devem consultar os respectivos professores antes da matrícula.” é configurada pela variável *class\_schedule\_text* descrita na Seção 4.2.


Uma importante modificação de estrutura que todos os relatórios sofreram foi a passagem de código de visualização (*view*) que estava em classes de controle (*controller*) para classes de visualização e de ajuda (*helper*). O uso de métodos de ajuda permitiu criar formatos padrões para todos os relatórios, com o mesmo estilo de tabelas, mesmo logo e mesmas cores, deixando toda a geração de relatórios padronizada. O uso de métodos de ajuda também facilitou a correção de problemas de quebra de página e de conteúdo sobrepondo o rodapé, que ocorriam na versão anterior.



O relatório de histórico foi modificado para atender exigências estabelecidas pela PROPPi. Dentre essas exigências estão: adição do nome do orientador, título da dissertação/tese, data de defesa, carga horária das disciplinas, banca avaliadora, estado expedidor da identidade, país e estado de naturalidade, área de concentração, conceito CAPES, resultado no exame de seleção e exame de línguas.


Muitos desses dados não eram armazenados no SAPOS, então, como parte da modificação do histórico, diversas modificações foram feitas: título da dissertação/tese, data de defesa e banca avaliadora adicionados em matrículas; local de expedição da identidade e local de naturalidade adicionados em alunos; carga horária adicionada em disciplinas; cidade adicionada em professores; e prova de inglês configurada como o exame de línguas. Para o resultado no exame de seleção, definiu-se o valor “Aprovado” para todas as matrículas cadastradas no SAPOS já que este seria o valor que iria aparecer no histórico, eliminando assim mais um campo que teria que ser preenchido.

A partir dessas modificações, o histórico teve de ser remodelado para ficar no mesmo estilo do histórico exigido pela PROPPi. A Figura 4.3 apresenta um histórico no novo estilo.

UNIVERSIDADE FEDERAL FLUMINENSE INSTITUTO DE COMPUTAÇÃO PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO					
HISTÓRICO ESCOLAR / PÓS-GRADUAÇÃO					
<b>ALUNO:</b> Hermione Granger <b>MATRÍCULA:</b> MH03 <b>NACIONALIDADE:</b> Inglaterra <b>DATA NASCIMENTO:</b> 19-09-1979 <b>IDENTIDADE:</b> 33.333.333-3 <b>ÓRGÃO EXPEDIDOR:</b> Dep. Bruxas <b>ESTADO EXPEDIDOR:</b> DBL <b>C.P.F.:</b> 333.333.333-33					
<b>CURSO:</b> Mestrado em Computação <b>ÁREA DE CONCENTRAÇÃO:</b> Herbologia <b>CONCEITO CAPES:</b> 5 <b>RESULTADO DO EXAME DE SELEÇÃO:</b> Aprovado <b>EXAME(S) DE LÍNGUA(S):</b> Aprovado em Prova de Inglês <b>MÊS/ANO DE INGRESSO NO CURSO:</b> Agosto/2012					
DISCIPLINAS CURSADAS COM APROVEITAMENTO					
CÓDIGO	DENOMINAÇÃO	NOTA / CONCEITO	NÚMERO DE CRÉDITOS	CARGA HORÁRIA	PERÍODO
HOG00100	Voo I	9.5	4	68h	2/2012
HOG00502	Controle dos Elementos	8.0	8	120h	2/2012
HOG00403	Popões	10.0	6	102h	2/2012
HOG00503	Tópicos Avançados em Fogo (A1)	7.0	4	68h	2/2012
HOG00101	Voo II	8.0	4	68h	1/2013
HOG00400	Plantas e Seus Usos	10.0	4	60h	1/2013
HOG00500	Levitação	7.0	4	68h	1/2013
HOG00302	Proteção Contra Encantamentos	6.0	8	130h	1/2013
HOG00402	Raízes	10.0	4	64h	2/2013
HOG00401	Sementes	10.0	6	102h	2/2013
<b>TOTAL</b>			52	850h	
<b>ORIENTADOR(ES):</b> Minerva McGonagall, Alastor Moody					
<b>MEMBROS DA BANCA EXAMINADORA / INSTITUIÇÃO DE ORIGEM:</b> Alastor Moody / Escola de Magia e Bruxaria de Hogwarts Severus Snape / Escola de Magia e Bruxaria de Hogwarts Remus Lupin / Academia de Magia Beauxbatons					
Niterói, 01 de Maio de 2014		Este documento só é válido sem rasuras, com selo da UFF e com a assinatura do Coordenador.			Pág. nº 1
		_____ ASSINATURA DO COORDENADOR			

**Figura 4.3: Histórico extraído do SAPOS 3**

Para deixar o boletim no mesmo formato que o novo histórico, o mesmo também foi remodelado. Diferentemente do histórico, o boletim é um relatório interno e não precisa mostrar conceito capes, exame de línguas e resultado do exame de seleção. Entretanto, o boletim precisa mostrar data de desligamento do curso com motivo, mostrar etapas concluídas, prorrogações, bolsas alocadas e separar o conjunto de disciplinas por período, indicando a situação em cada uma delas. A Figura 4.4 apresenta um boletim extraído do SAPOS 3 de uma matrícula sem prorrogações.

UNIVERSIDADE FEDERAL FLUMINENSE INSTITUTO DE COMPUTAÇÃO PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO							
BOLETIM ESCOLAR / PÓS-GRADUAÇÃO							
<b>ALUNO:</b> Hermione Granger <b>MATRÍCULA:</b> MH03 <b>NACIONALIDADE:</b> Inglaterra <b>DATA NASCIMENTO:</b> 19-09-1979 <b>IDENTIDADE:</b> 33.333.333-3 <b>ÓRGÃO EXPEDIDOR:</b> Dep. Bruxas <b>ESTADO EXPEDIDOR:</b> DBL <b>C.P.F.:</b> 333.333.333-33							
<b>CURSO:</b> Mestrado em Computação <b>ÁREA DE CONCENTRAÇÃO:</b> Herbologia <b>MÊS/ANO DE INGRESSO NO CURSO:</b> Agosto/2012 <b>MÊS/ANO DE DESLIGAMENTO DO CURSO:</b> Fevereiro/2014 <b>MOTIVO:</b> Titulação							
DISCIPLINAS CURSADAS COM APROVEITAMENTO							
CÓDIGO	DENOMINAÇÃO	NOTA / CONCEITO	NÚMERO DE CRÉDITOS	CARGA HORÁRIA	PERÍODO	SITUAÇÃO	
HOG00100	Voo I	9.5	4	68h	2/2012	Aprovado	
HOG00502	Controle dos Elementos	8.0	8	120h	2/2012	Aprovado	
HOG00403	Poções	10.0	6	102h	2/2012	Aprovado	
HOG00503	Tópicos Avançados em Fogo (A1)	7.0	4	68h	2/2012	Aprovado	
RESUMO DO PERÍODO		8.6	22	358h			
HOG00101	Voo II	8.0	4	68h	1/2013	Aprovado	
HOG00400	Plantas e Seus Usos	10.0	4	60h	1/2013	Aprovado	
HOG00500	Levitação	7.0	4	68h	1/2013	Aprovado	
HOG00302	Proteção Contra Encantamentos	6.0	8	130h	1/2013	Aprovado	
RESUMO DO PERÍODO		7.4	20	326h			
HOG00402	Raízes	10.0	4	64h	2/2013	Aprovado	
HOG00401	Sementes	10.0	6	102h	2/2013	Aprovado	
RESUMO DO PERÍODO		10.0	10	166h			
TOTAL		8.4	52	850h			
ORIENTADOR(ES): Minerva McGonagall, Alastor Moody							
MEMBROS DA BANCA EXAMINADORA / INSTITUIÇÃO DE ORIGEM: Alastor Moody / Escola de Magia e Bruxaria de Hogwarts Severus Snape / Escola de Magia e Bruxaria de Hogwarts Remus Lupin / Academia de Magia Beauxbatons							
Etapas obrigatórias concluídas							
Setembro/2012 Prova de Inglês							
Março/2013 Exame de Qualificação							
Fevereiro/2014 Pedido de Banca							
Bolsas							
Número	Agência	Data de Início	Data Limite	Encerramento			
F001	FAPERJ	Agosto/2012	Agosto/2014	Fevereiro/2014			
Niterói, 01 de Maio de 2014		Este documento só é válido sem rasuras, com selo da UFF e com a assinatura do Coordenador.				Pág. n° 1	
		ASSINATURA DO COORDENADOR					

**Figura 4.4: Boletim extraído do SAPOS 3**

Todos os outros relatórios também foram remodelados para o novo estilo. Entretanto, além do histórico e do boletim, os únicos que apresentaram diferenças nas informações mostradas foram a pauta de turmas e o relatório de matrículas. Na pauta de turmas, foi adicionada uma segunda página com o e-mail de todos os alunos da turma, como pode ser visto na Figura 4.5. Na primeira página da pauta, a coluna Situação foi removida. Já o relatório de matrículas passou a mostrar todos os critérios de busca utilizados na geração do relatório, como pode ser visto na Figura 4.6.

UNIVERSIDADE FEDERAL FLUMINENSE INSTITUTO DE COMPUTAÇÃO PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO			
LISTA DE E-MAILS			
NOME DA DISCIPLINA			SEMESTRE/ANO
Voo I			1º/2013
Nº	Matrícula	Nome do Aluno	E-mail do Aluno
1	DH01	Bill Weasley	bill.weasley@hogwarts.com
2	MH01	Draco Malfoy	draco.malfoy@hogwarts.com
3	MH02	Harry Potter	harry.potter@hogwarts.com
4	MH05	Ronald Weasley	ronald.weasley@hogwarts.com



**Figura 4.5: Última página da pauta extraída do SAPOS 3**

UNIVERSIDADE FEDERAL FLUMINENSE INSTITUTO DE COMPUTAÇÃO PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO			
Relatório de Matrículas			
Campo		Valor	
Nível		Doutorado	
Tipo de Matrícula		Regular	
Data de Admissão		Março/2013	
Ativo?		Todas	
Orientador		albus	
Realização de Etapa		Exame de Qualificação até 31 de Março de 2014	
Nome do Aluno	Número de Matrícula	Data de Admissão	Desligamento
Bill Weasley	DH01	2013-03-01	



**Figura 4.6: Relatório de matrículas extraído do SAPOS 3**

#### 4.4 MELHORIA NA VISUALIZAÇÃO E INTERAÇÃO

A terceira versão do SAPOS teve diversas melhorias na visualização e na interação, como adição de links de navegação em listas, redimensionamento de páginas, seleção de endereço, confirmação para sair de formulários, adição de linha do tempo em bolsas e outros. Esta seção apresenta cada uma dessas melhorias.

No SAPOS 2, cada linha das tabelas de entidades apresentava apenas textos com valores informados e links para editar, remover, visualizar e para baixar relatório(s). Quando uma coluna representava uma outra entidade, ficava apenas um texto descrevendo a entidade. Por exemplo, em matrícula, na coluna de aluno, aparecia apenas o nome do aluno. No SAPOS 3, ao invés de aparecer apenas o texto, colunas relacionadas apresentam um link para editar a entidade relacionada. Assim, ao clicar no nome do aluno

na lista de matrículas, aparece um formulário para editar as informações do aluno vinculado àquela matrícula.

Uma outra modificação nas linhas das tabelas foi a transformação dos links “Editar”, “Remover”, “Visualizar” e links de relatórios em ícones (*glyphicons*), a partir da utilização do *Font Awesome* (GANDY, 2013). Essa modificação permitiu mostrar mais informações nas tabelas. A Figura 4.7 apresenta uma parte da tabela de matrículas com essas alterações. Nessa figura, a tela de edição da aluna “Hermione Granger” está aberta.

Número de Matrícula	Nome do Aluno	Tipo de Matrícula	Nível	Data de Admissão	Desligamento	
DH01	Bill Weasley	Regular	Doutorado	Março-2013		
DH02	Charlie Weasley	Especial	Doutorado	Agosto-2013	Fevereiro-2014	
MH01	Draco Malfoy	Regular	Mestrado	Março-2013	Agosto-2013	
MH02	Harry Potter	Regular	Mestrado	Março-2013		
MH03	Hermione Granger	Regular	Mestrado	Agosto-2012	Fevereiro-2014	

**Nome**

**Sexo**

**Estado Civil**

**Data de Nascimento**

**Residência**

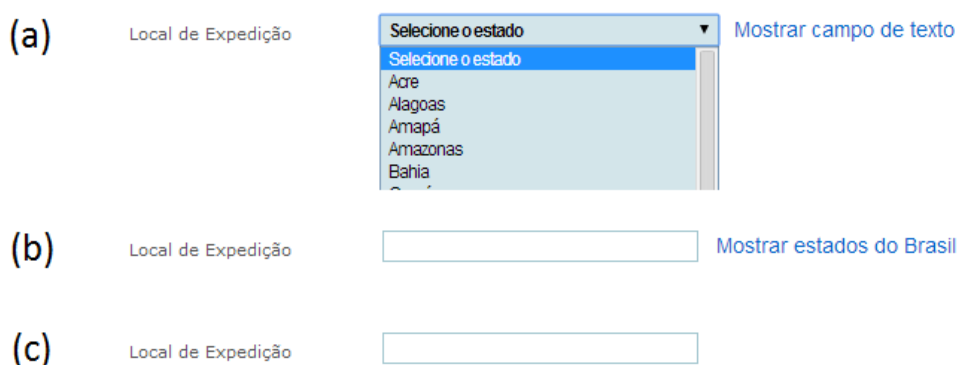
**Figura 4.7: Tabela de matrículas**

Outra modificação adicionada nesta versão do SAPOS foi a adição de uma confirmação para mudança de página sempre que um link que feche um formulário for clicado. Ou seja, se clicarem em “Professores”, ou em qualquer outro link do menu, aparecerá uma pergunta confirmando se é desejado fechar o formulário sem salvar.

Ainda na Figura 4.7, é possível ver o campo “Residência”. Na versão anterior do SAPOS, os campos “cidade”, “estado” e “país” eram independentes e o formulário permitia escolher o estado “Rio de Janeiro” com o país “Japão” ou a cidade “Niterói” com o estado “São Paulo”. No SAPOS 3, a lista de cidades só é carregada após escolher o estado, e a lista de estados só é carregada após escolher o país, evitando combinações incorretas. Além disso, a versão anterior armazenava os três campos, já a atual armazena apenas cidade.

Essa mesma alteração no campo residência também foi aplicada para o campo residência de professores e para o campo naturalidade de alunos, com a diferença de que o campo naturalidade armazena estado e cidade, para manter compatibilidade com os dados anteriores que armazenavam apenas estado de naturalidade.

Outra alteração no formulário de alunos e no formulário de professores foi o campo local de expedição da identidade, como pode ser visto na Figura 4.8.a. Este campo apresenta uma lista de estados do país configurado na variável *identity\_issuing\_country*, descrita na Seção 4.2. O campo possui um link para alternar a entrada entre a seleção de estados ou um texto qualquer, como pode ser visto na Figura 4.8.b. O texto pode ser usado para situações em que se deseja cadastrar um passaporte, ou algum outro documento que não seja a identidade. Quando a variável *identity\_issuing\_country* é configurada como cadeia de caracteres vazia, o campo local de expedição mostra apenas a entrada de texto, não permitindo alterar para a listagem de estados, como pode ser visto na Figura 4.8.c.

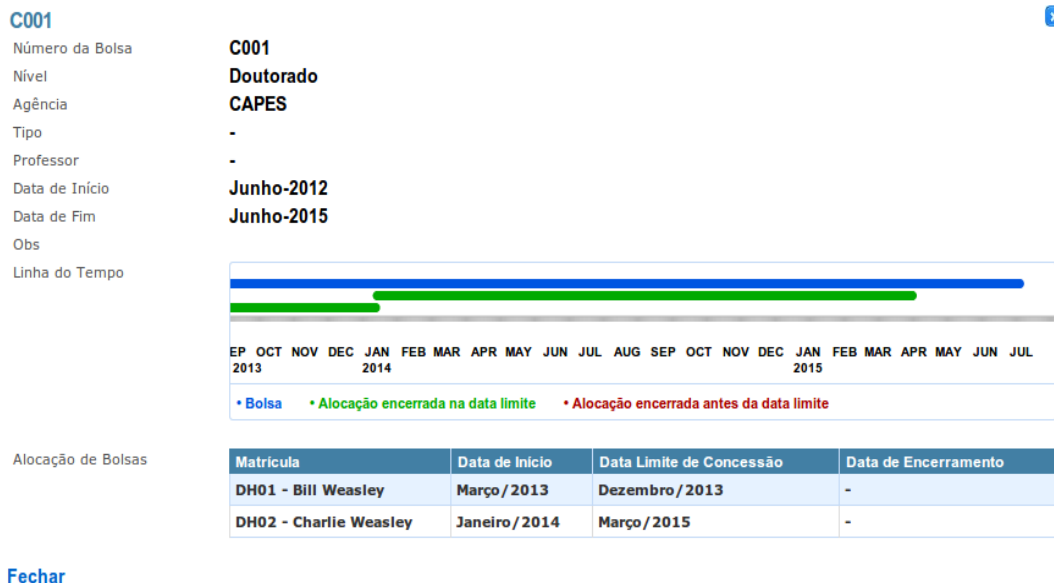


**Figura 4.8: Campo Local de Expedição**

O controle de bolsas também sofreu algumas alterações. A página de edição da bolsa passou a permitir o cadastro de alocações e a página de visualização ganhou uma linha do tempo e uma tabela com todas as alocações. Anteriormente, para criar uma alocação de bolsa, era necessário abrir a página de alocações e criar uma nova alocação, associando-a à bolsa; e para descobrir quais alocações estavam associadas a uma bolsa, era necessário realizar uma busca em alocações.

A Figura 4.9 apresenta a página de visualização de uma bolsa com a linha do tempo e com a tabela de alocações. A linha do tempo foi criada com a biblioteca *chronoline* (LEUNG et al., 2013), com o objetivo de observar de maneira simples quais alocações terminaram antes do tempo e em que momentos uma determinada bolsa esteve sem alocação.

Na página de prorrogações, foi adicionada uma coluna validade, indicando qual é a validade de cada etapa após a prorrogação, como pode ser visto na Figura 4.10. Essa validade é calculada a partir do prazo da etapa, da data de matrícula e da duração da prorrogação concedida.



**Figura 4.9: Visualização de Bolsa**



**Figura 4.10: Página de Prorrogações**

Além de todas essas mudanças na interface, uma mudança que ocorreu em diversas partes do sistema foi a transformação de campos de seleção de entidades em campos criados pelo *recordselect* (IVY; CAMBRA, 2013). Um campo do *recordselect* permite escrever e auto completa o resultado da busca com a entidade desejada. Muitas partes do sistema já usavam este tipo de campo anteriormente, essa alteração na interface serviu para padronizar os campos e para melhorar a interação.

## 4.5 DISCUSSÃO

Neste capítulo, vimos quais foram as principais alterações não relacionadas a notificações no SAPOS 3. Vimos o sistema de registro de mudanças (*log*), que permitiu uma maior confiabilidade na alteração de dados. Foi adicionado um sistema de variáveis armazenadas no banco que podem ser configuradas de acordo com cada Programa de Pós-

Graduação. Observamos também as mudanças realizadas no sistema como um todo para usar o formato de histórico da PROPPi e as mudanças realizadas nos outros relatórios, seja para adicionar mais informações, ou para padronizar o estilo. Por fim, observamos as diversas melhorias na interface e na interação que facilitaram o uso e visualização do sistema.

A terceira versão do SAPOS também recebeu outras mudanças. Em matrícula, foi adicionada uma busca por disciplina, ano e semestre que permite buscar quais alunos cursaram determinada disciplina em determinado ano ou semestre, onde o uso do ano e do semestre é opcional, sendo possível buscar quais alunos cursaram determinada disciplina. Diversas manutenções corretivas foram feitas, corrigindo problemas na busca de alocação de bolsas; na busca de orientações; na busca de matrículas ativas; no uso de *assets*; no CSS; e no armazenamento de datas de fim, que antes eram armazenadas como início do mês e passaram a ser armazenadas como o último dia do mês.

Além disso, foram adicionadas diversas novas validações para as entidades e algumas mensagens de erro geradas pelo *Rails* para validações comuns que não estavam traduzidas, foram traduzidas. Os campos de cadastro e edição de matrícula foram reordenados. Algumas consultas realizadas no sistema, que antes estavam escritas em SQL compatível com o *MySQL*, foram reescritas usando o mapeamento objeto-relacional (*ORM*) do *Rails* para funcionarem tanto no *MySQL*, em produção, quanto no *SQLite*, em desenvolvimento.

Com isso, vimos a maior parte das novas funcionalidades e correções que foram feitas no desenvolvimento do SAPOS 3, com o objetivo de atender as necessidades do PGC.

## CAPÍTULO 5 – CONCLUSÃO

Este trabalho teve como principal motivação a introdução do sistema de notificações automatizadas ao SAPOS, o que facilitou o trabalho de gestão do PGC, que precisava buscar a situação de cada aluno manualmente e notificá-lo caso necessário. No entanto, esta não foi a única melhoria que o este trabalho introduziu ao SAPOS – outras melhorias também foram adicionadas: melhorias na navegação e apresentação do sistema, a criação do sistema de log, criação do sistema de variáveis customizáveis, melhorias na geração de relatórios, e padronização do boletim e histórico escolar de acordo com as regras da PROPPi.

As melhorias introduzidas ao SAPOS através deste trabalho visavam facilitar a gestão das informações do PGC, suprimindo necessidades que haviam sido notadas a partir de trabalhos anteriores. Acreditamos que com a versão atual do SAPOS (3.3.5) temos um sistema robusto e capaz de suprir grande parte das necessidades de gestão de um Programa de Pós-Graduação.

Através de uma avaliação da situação atual percebemos que ainda existem melhorias que podem ser introduzidas ao sistema em futuros trabalhos. Uma delas é a introdução de consultas complexas, que seriam consultas que não são simples de implementar em uma interface para o usuário, mas que podem ser implementadas através de consultas em *Structured Query Language* (SQL). Esta consulta seria criada e teria um nome, um conjunto de parâmetros e uma instrução SQL atribuídas a ela, para que fosse salva no sistema para usos futuros.

Outros possíveis trabalhos futuros poderiam também fazer o armazenamento da foto do aluno, a generalização do SAPOS para que este pudesse ser utilizado em qualquer Programa de Pós-Graduação de diferentes universidades sem necessidade de adaptações no código, a criação de uma página para controlar e realizar releases e replicações do banco de dados de produção para homologação, e, por fim, um estudo focado na interação humano computador com o objetivo de avaliar e melhorar a usabilidade e interação.

As sugestões de trabalhos futuros apresentados nos parágrafos anteriores e todo o código do SAPOS, que é um projeto de código aberto sob a licença MIT (MIT, 1988), podem ser encontrados em <https://github.com/gems-uff/sapos>. Por ser um projeto de código aberto, qualquer pessoa que desejar pode ajudar no desenvolvimento do sistema através de *Pull Requests* do *GitHub* (2014).



## REFERÊNCIAS BIBLIOGRÁFICAS

COGNITION. **Intro to cron.** Disponível em: <<http://www.unixgeeks.org/security/newbie/unix/cron-1.html>>. Acesso em: 28 fev. 2014.

COSTALES, B. et al. **sendmail**. 4th. ed. [s.l.] O'Reilly Media, 2007.

FERREIRA, R.; AMARO, T. **SAPOS - Sistema de Apoio à Pós-Graduação**. Monografia de Conclusão de Graduação—Niterói, Brazil: Ciência da Computação, IC-UFF, 2013.

GANDY, D. **Font Awesome**. Disponível em: <<http://fontawesome.github.io/Font-Awesome/>>. Acesso em: 1 abr. 2014.

GITHUB. **Using pull requests**. Disponível em: <<https://help.github.com/articles/using-pull-requests>>. Acesso em: 1 maio. 2014.

HANSSON, D. **Active Record Basics — Ruby on Rails Guides**. Disponível em: <[http://guides.rubyonrails.org/active\\_record\\_basics.html](http://guides.rubyonrails.org/active_record_basics.html)>. Acesso em: 28 fev. 2014.

HAVERBEKE, M. **CodeMirror**. Disponível em: <<http://codemirror.net/>>. Acesso em: 28 fev. 2014.

IVY, L.; CAMBRA, S. **recordselect**. Disponível em: <<https://github.com/scambra/recordselect>>. Acesso em: 1 abr. 2014.

KRASNER, G. E.; POPE, S. T. A description of the model-view-controller user interface paradigm in the smalltalk-80 system. **Journal of object oriented programming**, v. 1, n. 3, p. 26–49, 1988.

LEUNG, K. et al. **Chronoline.js**. Disponível em: <<http://stoicloofah.github.io/chronoline.js/>>. Acesso em: 1 abr. 2014.

MATSUMOTO, Y. **String Format (Ruby 1.9.2)**. Disponível em: <<http://www.ruby-doc.org/core-1.9.2/String.html#method-i-25>>. Acesso em: 28 fev. 2014a.

MATSUMOTO, Y. **ERB (Ruby 1.9.2)**. Disponível em: <<http://ruby-doc.org/stdlib-1.9.2/libdoc/erb/rdoc/ERB.html>>. Acesso em: 28 fev. 2014b.

MATSUMOTO, Y. **Time strftime (Ruby 1.9.2)**. Disponível em: <<http://www.ruby-doc.org/core-1.9.2/Time.html#method-i-strftime>>. Acesso em: 28 fev. 2014c.

METTRAUX, J. **Rufus-Scheduler**. Disponível em: <<https://github.com/jmettraux/rufus-scheduler>>. Acesso em: 28 fev. 2014.

MIT. **The MIT License**. Disponível em: <<http://opensource.org/licenses/MIT>>. Acesso em: 1 maio. 2014.

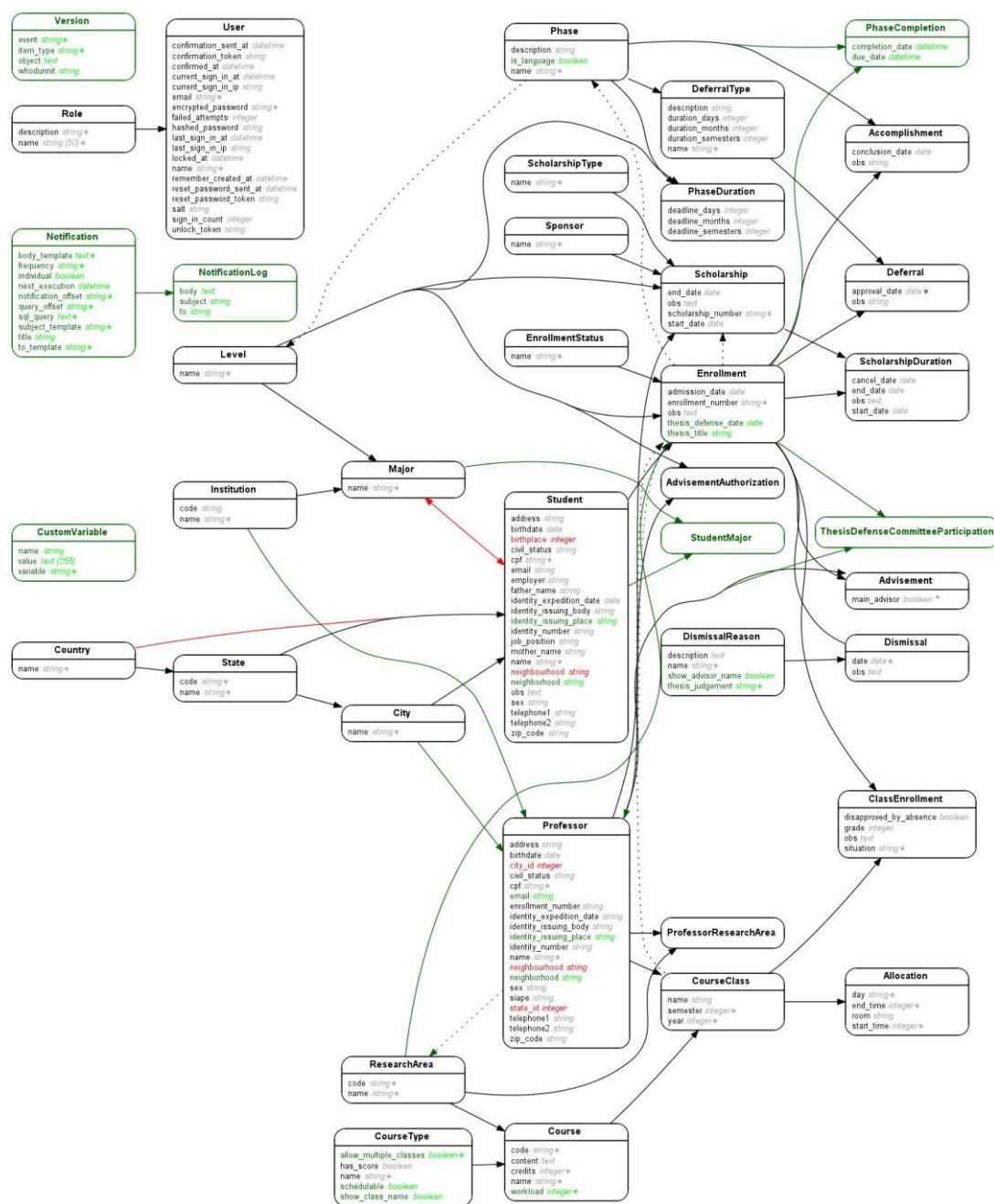
SCHETTINO, B. **SAPOS 2: Gestão de Disciplinas, Etapas e Credenciamentos no Sistema de Apoio à Pós-Graduação**. Monografia de Conclusão de Graduação—Niterói, Brazil: Ciência da Computação, IC-UFF, 2013.

STEWART, A. **paper\_trail**. Disponível em: <[https://github.com/airblade/paper\\_trail](https://github.com/airblade/paper_trail)>. Acesso em: 1 abr. 2014.

WHITE, R. **Active Scaffold**. Disponível em: <[https://github.com/activescaffold/active\\_scaffold](https://github.com/activescaffold/active_scaffold)>. Acesso em: 28 fev. 2014.

## APÊNDICE A – ALTERAÇÕES NO MODELO DE CLASSES DO SAPOS 3 EM RELAÇÃO AO SAPOS 2

No diagrama a seguir representamos as alterações ao modelo de classes do SAPOS 3 em relação ao SAPOS 2, as novidades adicionadas ao SAPOS 3 estão marcadas em verde, e em vermelho estão os elementos do sistema que se tornaram desnecessários e portanto foram removidos.



## ANEXO A - NOTIFICAÇÕES PROGRAMADAS CADASTRADAS

Atualmente, existem 23 notificações programadas cadastradas no servidor da Pós-Graduação em Computação da UFF, como pode ser visto a seguir.

1 - ALUNOS: Etapas vencidas ou a vencer em 1 mês	
	<b>Consulta:</b> SELECT students.email AS email, students.name AS name, phases.name AS phase_name, due_date AS due_date FROM phase_completions INNER JOIN enrollments ON enrollments.id = phase_completions.enrollment_id INNER JOIN students ON students.id = enrollments.student_id INNER JOIN phases ON phases.id = phase_completions.phase_id WHERE due_date <= {query_date} AND enrollments.id NOT IN (SELECT dismissals.enrollment_id from dismissals) AND enrollments.enrollment_status_id = 2 AND completion_date IS NULL /* Se fase for artigo A1 só vale para alunos que entraram depois de 2011.1 */ AND (phases.id <> 5 OR enrollments.admission_date >= '2011-03-01')
<b>Frequência:</b>	<b>(Um e-mail para cada resultado)</b>
Mensal	<b>Destinatário:</b> <%= email %>
	<b>Assunto:</b> Prazo para realização de <%= phase_name %>
<b>Desloc. de Notificação:</b>	<b>Corpo:</b>
-20	<%= name %>
<b>Desloc. de Consulta:</b>	<p>Informamos que falta pouco mais de um mês para o vencimento da etapa &lt;%= phase_name %&gt;. Para maiores informações, consulte as regras disponíveis em nosso site.</p> <p>Caso você já tenha realizado os procedimentos para o cumprimento dessa etapa, favor desconsiderar essa mensagem.</p> <p>Regras para Pedidos de Banca:  <a href="http://www.ic.uff.br/images/documentos/pos_graduacao/regras_e_procedimentos/resolucoes_do_colegiado/regras_de_prorrogacao_e_prazos_maximos_v2.pdf">http://www.ic.uff.br/images/documentos/pos_graduacao/regras_e_procedimentos/resolucoes_do_colegiado/regras_de_prorrogacao_e_prazos_maximos_v2.pdf</a></p> <p>Regras para Exame de Qualificação:  <a href="http://www.ic.uff.br/images/documentos/pos_graduacao/regras_e_procedimentos/resolucoes_do_colegiado/regras_de_prorrogacao_e_prazo_maximo_exam_qualificacao_v2.pdf">http://www.ic.uff.br/images/documentos/pos_graduacao/regras_e_procedimentos/resolucoes_do_colegiado/regras_de_prorrogacao_e_prazo_maximo_exam_qualificacao_v2.pdf</a></p> <p>Regras para Prova de Inglês:  <a href="http://www.ic.uff.br/images/documentos/pos_graduacao/regras_e_procedimentos/resolucoes_do_colegiado/criterios_regras_e_procedimentos_v5.pdf">http://www.ic.uff.br/images/documentos/pos_graduacao/regras_e_procedimentos/resolucoes_do_colegiado/criterios_regras_e_procedimentos_v5.pdf</a>  (itens 4 e 5).</p>
0	

2 - CORD: Alunos ainda não desligados	
<b>Frequência:</b> Mensal	<b>Consulta:</b> SELECT students.email AS email, students.name AS name, due_date AS due_date FROM phase_completions INNER JOIN enrollments ON enrollments.id = phase_completions.enrollment_id INNER JOIN students ON students.id = enrollments.student_id WHERE completion_date <= %{query_date} AND enrollments.id NOT IN (SELECT dismissals.enrollment_id from dismissals) AND phase_completions.phase_id = 1 AND enrollments.enrollment_status_id = 2
<b>Desloc. de Notificação:</b> 5	<b>(Um e-mail com todos os resultados)</b>
<b>Desloc. de Consulta:</b> -45	<b>Destinatário:</b> <%= User.where(:role_id => 2).map(&:email).join(';') %> <b>Assunto:</b> Alunos ainda não desligados <b>Corpo:</b> Prezado Coordenador,  Informamos que os seguintes alunos pediram banca há mais de 45 dias, mas ainda não foram desligados no SAPOS:  <% records.each do  record  %> - <%= record['name'] %> (<%= record['email'] %>) <% end %>

3 - CORD: Número de alunos ativos	
<b>Frequência:</b> Semestral	<b>Consulta:</b> SELECT count(*) AS num_alunos, levels.name AS level FROM enrollments, levels WHERE enrollments.id NOT IN (SELECT enrollment_id FROM dismissals WHERE date < %{query_date}) AND admission_date < %{query_date} AND enrollment_status_id = 2 AND enrollments.level_id = levels.id
<b>Desloc. de Notificação:</b> -3M	GROUP BY level_id <b>(Um e-mail com todos os resultados)</b>
<b>Desloc. de Consulta:</b> -3M	<b>Destinatário:</b> <%= User.where(:role_id => 2).map(&:email).join(';') %> <b>Assunto:</b> Total de alunos ativos <b>Corpo:</b> Prezado Coordenador  Informamos que o número de alunos ativos do PGC é:  <% records.each do  record  %> - <%= record['level'] %> (<%= record['num_alunos'] %>) <% end %>

4 - CORD: Alunos a serem jubilados por CR < 6 (DEZ)	
<b>Frequência:</b> Anual  <b>Desloc. de Notificação:</b> -15  <b>Desloc. de Consulta:</b> 0	<b>Consulta:</b> SELECT s.name, e.enrollment_number, cc.year, cc.semester, (SUM(ce.grade * c.credits)/SUM(c.credits))/10 AS cr_perodo FROM students s, enrollments e, class_enrollments ce, courses c, course_types ct, course_classes cc WHERE s.id = e.student_id AND e.id NOT IN (SELECT enrollment_id FROM dismissals) AND e.enrollment_status_id = 2 AND ce.enrollment_id = e.id AND ct.has_score = 1 AND ct.id = c.course_type_id AND c.id = cc.course_id AND cc.id = ce.course_class_id GROUP BY s.name, e.enrollment_number, cc.year, cc.semester HAVING SUM(ce.grade * c.credits)/SUM(c.credits) < 60 ORDER BY enrollment_number <b>(Um e-mail com todos os resultados)</b> <b>Destinatário:</b> <%= User.where(:role_id => 2).map(&:email).join(';') %> <b>Assunto:</b> Alunos a serem jubilados por CR < 6 <b>Corpo:</b> Prezado Coordenador  Abaixo estão listados os alunos a serem jubilados por CR menor do que 6.  <% records.each do  record  %> - <%= record['enrollment_number'] %> <%= record['name'] %> -- CR <%= record['cr_perodo'] %> <% end %>

5 - CORD: Alunos a serem jubilados por CR < 6 (JUL)	
<b>Frequência:</b> Anual  <b>Desloc. de Notificação:</b> -5M10d  <b>Desloc. de Consulta:</b> 30	<b>Consulta:</b> Igual a Notificação 4 <b>(Um e-mail com todos os resultados)</b> <b>Destinatário:</b> Igual a Notificação 4 <b>Assunto:</b> Igual a Notificação 4 <b>Corpo:</b> Igual a Notificação 4

6 – CORD: Alunos a serem jubilados por CR < 7 em dois períodos consecutivos (DEZ)	
<p><b>Frequência:</b> Anual</p> <p><b>Desloc. de Notificação:</b> -15</p> <p><b>Desloc. de Consulta:</b> 0</p>	<p><b>Consulta:</b></p> <pre> SELECT s.name,        enrollment_number,        cc.year,        cc.semester,        (SUM(ce.grade * c.credits)/SUM(c.credits))/10 AS cr_periodo FROM students s, enrollments e, class_enrollments ce,      courses c, course_types ct, course_classes cc WHERE s.id = e.student_id       AND e.id NOT IN (SELECT enrollment_id FROM dismissals)       AND e.enrollment_status_id = 2       AND ce.enrollment_id = e.id       AND ct.has_score = 1       AND ct.id = c.course_type_id       AND c.id = cc.course_id       AND cc.id = ce.course_class_id       AND cc.year = %{this_semester_year}       AND cc.semester = %{this_semester_number} /* e também teve CR menor do que 7 no semestre anterior */       AND e.id IN (SELECT e.id                   FROM enrollments e, class_enrollments ce, courses c,                        course_types ct, course_classes cc                   WHERE e.id NOT IN (SELECT enrollment_id FROM dismissals)                         AND e.enrollment_status_id = 2                         AND ce.enrollment_id = e.id                         AND ct.has_score = 1                         AND ct.id = c.course_type_id                         AND c.id = cc.course_id                         AND cc.id = ce.course_class_id                         AND cc.year = %{last_semester_year}                         AND cc.semester = %{last_semester_number}                   GROUP BY e.id                   HAVING SUM(ce.grade * c.credits)/SUM(c.credits) &lt; 70) GROUP BY s.name, e.enrollment_number, cc.year, cc.semester HAVING SUM(ce.grade * c.credits)/SUM(c.credits) &lt; 70 ORDER BY enrollment_number </pre> <p><b>(Um e-mail com todos os resultados)</b></p> <p><b>Destinatário:</b> &lt;%= User.where(:role_id =&gt; 2).map(&amp;:email).join(';') %&gt;</p> <p><b>Assunto:</b> Alunos a serem jubilados por CR &lt; 7 em dois períodos consecutivos</p> <p><b>Corpo:</b></p> <p>Prezado Coordenador</p> <p>Abaixo estão listados os alunos a serem jubilados por CR menor do que 7 em dois períodos consecutivos.</p> <pre> &lt;% records.each do  record  %&gt;   - &lt;%= record['enrollment_number'] %&gt; &lt;%= record['name'] %&gt; -- CR &lt;%= record['cr_periodo'] %&gt; &lt;% end %&gt; </pre>

7 – CORD: Alunos a serem jubilados por CR < 7 em dois períodos consecutivos (JUL)	
<b>Frequência:</b> Anual	<b>Consulta:</b> Igual a Notificação 6
	<b>(Um e-mail com todos os resultados)</b>
<b>Desloc. de Notificação:</b> -5M15d	<b>Destinatário:</b> Igual a Notificação 6
	<b>Assunto:</b> Igual a Notificação 6
<b>Desloc. de Consulta:</b> 0	<b>Corpo:</b> Igual a Notificação 6

8 – CORD: Alunos a serem jubilados por 2 ou mais reprovações (DEZ)	
<b>Frequência:</b> Anual	<b>Consulta:</b> SELECT s.name, enrollment_number, COUNT(*) AS num_reprovacoes FROM students s, enrollments e, class_enrollments ce, courses c, course_types ct, course_classes cc WHERE s.id = e.student_id AND e.id NOT IN (SELECT enrollment_id FROM dismissals) AND e.enrollment_status_id = 2 AND ce.enrollment_id = e.id AND ct.id = c.course_type_id AND c.id = cc.course_id AND cc.id = ce.course_class_id AND ce.situation = "Reprovado" GROUP BY s.name, e.enrollment_number HAVING NUM_REPROVACOES >= 2 ORDER BY NUM_REPROVACOES DESC, enrollment_number
	<b>(Um e-mail com todos os resultados)</b>
<b>Desloc. de Notificação:</b> -15d	<b>Destinatário:</b> <%= User.where(:role_id => 2).map(&:email).join(';') %>
	<b>Assunto:</b> Alunos a serem jubilados por 2 ou mais reprovações
<b>Desloc. de Consulta:</b> 0	<b>Corpo:</b> Prezado Coordenador  Abaixo segue a lista de alunos a serem jubilados por terem 2 ou mais reprovações.  <% records.each do  record  %> - <%= record['enrollment_number'] %> <%= record['name'] %> -- Num Reprovações: <%= record['num_reprovacoes'] %>  <% end %>

9 – CORD: Alunos a serem jubilados por 2 ou mais reprovações (JUL)	
<b>Frequência:</b> Anual	<b>Consulta:</b> Igual a Notificação 8
	<b>(Um e-mail com todos os resultados)</b>
<b>Desloc. de Notificação:</b> -5M15d	<b>Destinatário:</b> Igual a Notificação 8
	<b>Assunto:</b> Igual a Notificação 8
<b>Desloc. de Consulta:</b> 0	<b>Corpo:</b> Igual a Notificação 8



10 – PROFS: Vencimento de etapas em 1 mês	
	<p><b>Consulta:</b></p> <pre> SELECT professors.email AS email, professors.name AS prof_name,       students.name AS name, phases.name AS phase_name,       due_date AS due_date FROM phase_completions       INNER JOIN enrollments ON enrollments.id = phase_completions.enrollment_id       INNER JOIN students ON students.id = enrollments.student_id       INNER JOIN phases ON phases.id = phase_completions.phase_id       INNER JOIN advisements ON advisements.enrollment_id = enrollments.id       INNER JOIN professors ON professors.id = advisements.professor_id WHERE due_date&lt;=%{query_date} AND enrollments.id NOT IN (SELECT dismissals.enrollment_id from dismissals) AND enrollments.enrollment_status_id = 2 AND completion_date IS NULL /* Se fase for artigo A1 só vale para alunos que entraram depois de 2011.1 */ AND (phases.id &lt;&gt; 5 OR enrollments.admission_date &gt;= '2011-03-01')</pre>
	<b>(Um e-mail para cada resultado)</b>
<b>Frequência:</b>	<b>Destinatário:</b> <%= email %>
Mensal	<b>Assunto:</b> Prazo para realização de <%= phase_name %> de seu orientando <%= name %>
<b>Desloc. de Notificação:</b>	<b>Corpo:</b>
-20	<%= prof_name %>
<b>Desloc. de Consulta:</b>	<p>Informamos que falta pouco mais de um mês para o vencimento da etapa &lt;%= phase_name %&gt; de seu orientando &lt;%= name %&gt;. Para maiores informações, consulte as regras disponíveis em nosso site.</p> <p>Caso seu aluno já tenha realizado os procedimentos para o cumprimento dessa etapa, favor desconsiderar essa mensagem.</p> <p>Regras para Pedidos de Banca:  <a href="http://www.ic.uff.br/images/documentos/pos_graduacao/regras_e_procedimentos/resolucoes_do_colegiado/regras_de_prorrogacao_e_prazos_maximos_v2.pdf">http://www.ic.uff.br/images/documentos/pos_graduacao/regras_e_procedimentos/resolucoes_do_colegiado/regras_de_prorrogacao_e_prazos_maximos_v2.pdf</a></p> <p>Regras para Exame de Qualificação:  <a href="http://www.ic.uff.br/images/documentos/pos_graduacao/regras_e_procedimentos/resolucoes_do_colegiado/regras_de_prorrogacao_e_prazo_maximo_exam_qualificacao_v2.pdf">http://www.ic.uff.br/images/documentos/pos_graduacao/regras_e_procedimentos/resolucoes_do_colegiado/regras_de_prorrogacao_e_prazo_maximo_exam_qualificacao_v2.pdf</a></p> <p>Regras para Prova de Inglês:  <a href="http://www.ic.uff.br/images/documentos/pos_graduacao/regras_e_procedimentos/resolucoes_do_colegiado/criterios_regras_e_procedimentos_v5.pdf">http://www.ic.uff.br/images/documentos/pos_graduacao/regras_e_procedimentos/resolucoes_do_colegiado/criterios_regras_e_procedimentos_v5.pdf</a>  (itens 4 e 5).</p>
0	

11 – CORD: Avulsos cursando mais de 3 semestres	
<b>Frequência:</b> Semestral	<b>Consulta:</b> SELECT s.name AS name, s.email AS email, e.enrollment_number AS enrollment_number FROM students s, enrollments e, enrollment_statuses es WHERE s.id = e.student_id AND e.id NOT IN (SELECT enrollment_id FROM dismissals)  /* Se estiver inscrito em mais de 2 anos semestres distintos */ AND (SELECT COUNT(DISTINCT cc.year, cc.semester) FROM class_enrollments ce, course_classes cc WHERE e.id = enrollment_id AND cc.id = ce.course_class_id) > 2 AND es.id = e.enrollment_status_id AND es.name = "Avulso" ORDER BY s.name
<b>Desloc. de Notificação:</b> -15d	(Um e-mail com todos os resultados)
<b>Desloc. de Consulta:</b> 0	<b>Destinatário:</b> <%= User.where(:role_id => 2).map(&:email).join(',') %> <b>Assunto:</b> Alunos avulsos cursando mais do que dois semestres <b>Corpo:</b> Prezado Coordenador  Os alunos avulsos listados abaixo estão inscritos em disciplinas como avulso em mais do que dois semestres, contrariando a regra vigente.  <% records.each do  record  %> - <%= record['name'] %> (<%= record['email'] %>) - <%= record['enrollment_number'] %> <% end %>

12 – ALUNOS: Fim da validade do trancamento	
<b>Frequência:</b> Semestral	<b>Consulta:</b> SELECT s.name, s.email FROM students s, enrollments e WHERE s.id = e.student_id AND enrollment_status_id = 3 /* Está trancado no momento */ AND e.id NOT IN (SELECT dismissals.enrollment_id from dismissals) /* Tem uma prorrogação do tipo trancamento concedida antes da data de execução da consulta */ AND e.id IN (SELECT enrollment_id FROM deferrals WHERE deferral_type_id = 9 AND approval_date < %{query_date})
<b>Desloc. de Notificação:</b> -10	(Um e-mail para cada resultado)
<b>Desloc. de Consulta:</b> 0	<b>Destinatário:</b> <%= email %> <b>Assunto:</b> Validade do trancamento terminou <b>Corpo:</b> <%= name %>  Informamos que a validade do seu trancamento termina no final do mês. Favor entrar em contato com a secretaria para regularizar sua situação.

13 – CORD: Alunos de doutorado que se matricularam em Proposta de Tese mas não têm 2 básicas com nota >= 7	
<p><b>Frequência:</b> Semestral</p> <p><b>Desloc. de Notificação:</b> -10d</p> <p><b>Desloc. de Consulta:</b> 0</p>	<p><b>Consulta:</b>  SELECT s.name, enrollment_number, admission_date, e.obs  FROM students s, enrollments e, enrollment_statuses es, levels l  WHERE s.id = e.student_id  AND e.id NOT IN (SELECT enrollment_id FROM dismissals)  AND es.id = e.enrollment_status_id  AND es.name = "Regular"  AND l.name = "Doutorado"  AND e.level_id=l.id  AND (e.obs NOT LIKE "%%BASICAS &gt; 7 OK%%" OR e.obs IS NULL)  /* Ja se matriculou em Tese -&gt; como não temos oficialmente Proposta de Tese, estamos usando Tese */  AND e.id IN (SELECT en.id FROM enrollments en, class_enrollments ce, courses c, course_classes cc  WHERE c.id = cc.course_id  AND cc.id = ce.course_class_id  AND en.id = ce.enrollment_id  AND c.id = 52)  AND s.id NOT IN (SELECT st.id  FROM students st, enrollments en, class_enrollments ce, courses c, course_types ct, course_classes cc  WHERE ct.id = c.course_type_id  AND st.id = en.student_id  AND c.id = cc.course_id  AND cc.id = ce.course_class_id  AND en.id = ce.enrollment_id  AND (ce.situation = "Aprovado")  AND ct.id = 4  AND ce.grade &gt;= 70  AND en.enrollment_status_id = 2  GROUP BY enrollment_number  HAVING count(*) &gt;= 2)  ORDER BY s.name</p> <p><b>(Um e-mail com todos os resultados)</b></p> <p><b>Destinatário:</b> &lt;%= User.where(:role_id =&gt; 2).map(&amp;:email).join(';') %&gt;</p> <p><b>Assunto:</b> Alunos de doutorado que se matricularam em Proposta de Tese mas não têm 2 básicas com nota &gt;= 7</p> <p><b>Corpo:</b>  Prezado Coordenador</p> <p>Os alunos abaixo inscreveram-se em Proposta de Tese mas não cumpriram o requisito exigido para defesa de Exame de Qualificação (duas básicas do curso com nota maior ou igual a 7):</p> <p>&lt;% records.each do  record  %&gt;  - &lt;%= record['name'] %&gt; - &lt;%= record['enrollment_number'] %&gt; (&lt;%= record['obs'] %&gt;)  &lt;% end %&gt;</p>

14 – CORD: Fim da validade de trancamentos	
<b>Frequência:</b> Semestral  <b>Desloc. de Notificação:</b> -9  <b>Desloc. de Consulta:</b> 0	<p><b>Consulta:</b></p> <pre> SELECT s.name,        s.email,        e.enrollment_number FROM students s,        enrollments e WHERE s.id = e.student_id /* Está trancado no momento */ AND enrollment_status_id = 3 AND e.id NOT IN (SELECT dismissals.enrollment_id                  FROM dismissals) /* Tem uma prorrogação do tipo trancamento concedida antes da data de execução da consulta */ AND e.id IN (SELECT enrollment_id              FROM deferrals              WHERE deferral_type_id = 9              AND approval_date &lt; %{query_date}) </pre> <p><b>(Um e-mail com todos os resultados)</b></p> <p><b>Destinatário:</b> &lt;%= User.where(:role_id =&gt; 2).map(&amp;:email).join(';') %&gt;; secretaria.pos@ic.uff.br</p> <p><b>Assunto:</b> Alunos cujo trancamento expira no final do mês</p> <p><b>Corpo:</b></p> <p>Prezado Coordenador</p> <p>Informamos que a validade do trancamento dos alunos abaixo relacionados termina no final do mês.</p> <p>Aproveitamos para informar que todos eles foram notificados de que devem entrar em contato com a secretaria para regularizar sua situação.</p> <p>&lt;% records.each do  record  %&gt;  - &lt;%= record['name'] %&gt; - &lt;%= record['enrollment_number'] %&gt; - &lt;%= record['email'] %&gt;  &lt;% end %&gt;</p>

15 – CORD: Alunos que não se inscreveram em seminários	
<b>Frequência:</b> Semestral  <b>Desloc. de Notificação:</b> 5  <b>Desloc. de Consulta:</b> 0	<p><b>Consulta:</b></p> <pre> SELECT s.name AS name,        e.enrollment_number AS enrollment_number,        e.admission_date AS date FROM students s, enrollments e WHERE s.id = e.student_id AND e.id NOT IN (   SELECT enrollment_id   FROM dismissals ) AND e.id NOT IN (   SELECT enrollment_ID   FROM class_enrollments ce, course_classes cc, courses c   WHERE ce.course_class_id = cc.id   AND c.id = cc.course_id   AND c.course_type_id = 1 ) AND e.enrollment_status_id = 2 ORDER BY admission_date </pre> <p><b>(Um e-mail com todos os resultados)</b></p> <p><b>Destinatário:</b> &lt;%= User.where(:role_id =&gt; 2).map(&amp;:email).join(';') %&gt;</p> <p><b>Assunto:</b> Alunos que não se inscreveram em Seminários</p> <p><b>Corpo:</b></p> <p>Prezado Coordenador,</p> <p>Informamos que os alunos abaixo nunca se inscreveram em Seminários:</p> <p>&lt;% records.each do  record  %&gt;      - &lt;%= record['name'] %&gt; (&lt;%= record['enrollment_number'] %&gt;), data de    admissão: &lt;%= localize(record['date'], :monthyear) %&gt;    &lt;% end %&gt;</p> <p>Cabe ressaltar que disciplinas de alunos antigos não foram completamente migradas para o SAPOS.</p>

16 – CORD: Alunos que não se matricularam	
<b>Frequência:</b> Semestral  <b>Desloc. de Notificação:</b> 5  <b>Desloc. de Consulta:</b> 0	<p><b>Consulta:</b></p> <pre> SELECT s.name AS name,        s.email AS email,        e.enrollment_number AS enrollment_number,        e.admission_date AS date,        es.name AS status FROM students s,      enrollments e,      enrollment_statuses es WHERE s.id = e.student_id AND e.id NOT IN (SELECT enrollment_id                  FROM dismissals) AND e.id NOT IN (SELECT enrollment_ID                  FROM class_enrollments ce, course_classes cc                  WHERE year = 2014 and semester = 1                  AND ce.course_class_id = cc.id) /* se não tiver pedido banca ainda */ AND e.id NOT IN (SELECT enrollment_id                  FROM accomplishments                  WHERE phase_id = 1) AND es.id = e.enrollment_status_id /* considera matrícula regular ou trancada */ AND (es.id = 2 OR es.id = 3) ORDER BY date </pre> <p><b>(Um e-mail com todos os resultados)</b></p> <p><b>Destinatário:</b> &lt;%= User.where(:role_id =&gt; 2).map(&amp;:email).join(';') %&gt;</p> <p><b>Assunto:</b> Alunos que não se matricularam</p> <p><b>Corpo:</b></p> <p>Prezado Coordenador,</p> <p>Informamos que os alunos abaixo não se matricularam:</p> <pre> &lt;% records.each do  record  %&gt;   - &lt;%= record['name'] %&gt; (&lt;%= record['enrollment_number'] %&gt;), status da matrícula: (&lt;%= record['status'] %&gt;), data de admissão: &lt;%= localize(record['date'], :monthyear) %&gt; &lt;% end %&gt; </pre>

17 – CORD: Alunos de mestrado com mais de 12 créditos em EO ou Tópicos	
<b>Frequência:</b> Semestral  <b>Desloc. de Notificação:</b> 5  <b>Desloc. de Consulta:</b> 0	<b>Consulta:</b> <pre> SELECT s.name AS name,        e.enrollment_number AS enrollment_number,        SUM(c.credits) AS total_credits FROM students s,      enrollments e,      `class_enrollments` ce,      courses c,      course_types ct,      course_classes cc WHERE s.id = e.student_id AND e.id NOT IN (SELECT enrollment_id                  FROM dismissals) AND e.enrollment_status_id = 2 AND e.level_id=3 AND ce.enrollment_id = e.id AND ct.has_score = 1 AND ct.id = c.course_type_id /* Topicos ou Estudo Orientado */ AND (ct.id = 8 OR ct.id = 3) AND c.id = cc.course_id AND cc.id = ce.course_class_id GROUP BY s.name,          e.enrollment_number HAVING SUM(c.credits) &gt; 12 ORDER BY s.name </pre>
	<b>(Um e-mail com todos os resultados)</b>
	<b>Destinatário:</b> <%= User.where(:role_id => 2).map(&:email).join(';') %>
	<b>Assunto:</b> Alunos inscritos em mais de 12 créditos de EO e Tópicos
	<b>Corpo:</b> Prezado Coordenador,  Informamos que os seguintes alunos de mestrado cursaram ou estão inscritos em mais de 12 créditos de disciplinas de Estudo Orientado ou Tópicos:  <pre> &lt;% records.each do  record  %&gt;   - &lt;%= record['name'] %&gt; (&lt;%= record['enrollment_number'] %&gt;), total de   créditos: &lt;%= record['total_credits'] %&gt; &lt;% end %&gt; </pre>
	Importante ressaltar que essa regra começou a ser checada automaticamente em Março de 2014.

18 – CORD: Alunos de mestrado que se matricularam em dissertação e não cumpriram os requisitos do curso (parte 1)	
<p><b>Frequência:</b> Semestral</p> <p><b>Desloc. de Notificação:</b> 5</p> <p><b>Desloc. de Consulta:</b> 0</p>	<p><b>Consulta:</b></p> <pre> SELECT s.name AS name, e.enrollment_number AS enrollment_number,        e.admission_date AS date FROM students s, enrollments e, enrollment_statuses es WHERE s.id = e.student_id AND e.id NOT IN (SELECT enrollment_id FROM dismissals) /* Aluno regular */ AND e.enrollment_status_id = 2 /* Aluno de mestrado */ AND e.level_id=3 /* Já se matriculou em dissertação */ AND e.id IN (SELECT enrollment_id              FROM class_enrollments ce, course_classes cc, courses c              WHERE ce.course_class_id = cc.id              AND c.course_type_id = 2              AND c.id = cc.course_id              ) AND /* Não cursou uma básica da área */ (e.id NOT IN (SELECT en.id              FROM enrollments en, class_enrollments ce, courses c,              course_types ct, course_classes cc              WHERE ct.id = c.course_type_id              AND c.research_area_id = en.research_area_id              AND c.id = cc.course_id              AND cc.id = ce.course_class_id              AND en.id = ce.enrollment_id              AND (ce.situation = "Aprovado")              AND ct.id = 5              GROUP BY enrollment_number              HAVING count(*) &gt;= 1) OR /* Não cursou duas básicas do curso */ e.id NOT IN (SELECT en.id             FROM enrollments en,             class_enrollments ce,             courses c,             course_types ct,             course_classes cc             WHERE en.id = ce.enrollment_id             AND ct.id = c.course_type_id AND c.id = cc.course_id             AND cc.id = ce.course_class_id             AND ce.situation = "Aprovado"             AND ct.id =4             GROUP BY enrollment_number             HAVING COUNT( * ) &gt;=2             ) ... Continua </pre>



18 – CORD: Alunos de mestrado que se matricularam em dissertação e não cumpriram os requisitos do curso (parte 2)	
<p><b>Frequência:</b> Semestral</p> <p><b>Desloc. de Notificação:</b> 5</p> <p><b>Desloc. de Consulta:</b> 0</p>	<pre> OR /* Não cursou Seminários */ e.id NOT IN (SELECT en.id              FROM enrollments en, class_enrollments ce, courses c, course_types ct,              course_classes cc              WHERE en.id = ce.enrollment_id              AND ct.id = c.course_type_id AND c.id = cc.course_id              AND cc.id = ce.course_class_id              AND ce.situation = "Aprovado" AND ct.id = 1              )  OR /* Não completou os créditos mas já se inscreveu em dissertação */ e.id IN (SELECT en.id         FROM enrollments en, class_enrollments ce, courses c, course_types ct,         course_classes cc         WHERE ct.has_score = 1         AND en.id = ce.enrollment_id AND cc.id = ce.course_class_id         AND c.id = cc.course_id AND ct.id = c.course_type_id         AND ce.situation = "Aprovado"         AND en.id IN (SELECT enrollment_id                      FROM class_enrollments ce, course_classes cc, courses c                      WHERE ce.course_class_id = cc.id                      AND c.course_type_id = 2                      AND c.id = cc.course_id                      )         ) GROUP BY en.id HAVING SUM( c.credits ) &lt; 32 ) GROUP BY s.name, e.enrollment_number, e.admission_date ORDER BY s.name </pre> <p><b>(Um e-mail com todos os resultados)</b></p> <p><b>Destinatário:</b> &lt;%= User.where(:role_id =&gt; 2).map(&amp;:email).join(';') %&gt;</p> <p><b>Assunto:</b> Alunos de mestrado que se inscreveram em dissertação mas não cumpriram os requisitos do curso</p> <p><b>Corpo:</b>          Prezado Coordenador,</p> <p>Informamos que os seguintes alunos de mestrado se inscreveram em dissertação mas não cumpriram pelo menos um dos requisitos do curso:</p> <ul style="list-style-type: none"> <li>- 1 básica da área</li> <li>- 2 básicas do curso</li> <li>- seminários</li> <li>- 32 créditos</li> </ul> <p>&lt;% records.each do  record  %&gt;              - &lt;%= record['name'] %&gt; (&lt;%= record['enrollment_number'] %&gt;), data de          admissão: (&lt;%= localize(record['date'], :monthyear) %&gt;          &lt;% end %&gt;</p>

19 – CORD: Alunos de doutorado com mais de 4 créditos em EO	
<b>Frequência:</b> Semestral  <b>Desloc. de Notificação:</b> 5  <b>Desloc. de Consulta:</b> 0	<p><b>Consulta:</b></p> <pre> SELECT s.name AS name,        e.enrollment_number AS enrollment_number,        SUM(c.credits) AS total_credits FROM students s, enrollments e, `class_enrollments` ce,        courses c, course_types ct, course_classes cc WHERE s.id = e.student_id AND e.id NOT IN (SELECT enrollment_id FROM dismissals) AND e.enrollment_status_id = 2 AND e.level_id=4 AND ce.enrollment_id = e.id AND ct.has_score = 1 AND ct.id = c.course_type_id /* Estudo Orientado */ AND ct.id = 8 AND c.id = cc.course_id AND cc.id = ce.course_class_id GROUP BY s.name, e.enrollment_number HAVING SUM(c.credits) &gt; 4 ORDER BY s.name </pre> <p><b>(Um e-mail com todos os resultados)</b></p> <p><b>Destinatário:</b> &lt;%= User.where(:role_id =&gt; 2).map(&amp;:email).join(';') %&gt;</p> <p><b>Assunto:</b> Alunos de doutorado com mais de 4 créditos em EO</p> <p><b>Corpo:</b></p> <p>Prezado Coordenador,</p> <p>Informamos que os seguintes alunos de doutorado cursaram ou estão inscritos em mais de 4 créditos de disciplinas de Estudo Orientado:</p> <pre> &lt;% records.each do  record  %&gt;   - &lt;%= record['name'] %&gt; (&lt;%= record['enrollment_number'] %&gt;), total de créditos: &lt;%= record['total_credits'] %&gt; &lt;% end %&gt; </pre> <p>Importante ressaltar que essa regra começou a ser checada automaticamente em Março de 2014.</p>

20 – CORD: Alunos de doutorado com mais de 8 créditos em disciplinas de Tópicos	
<b>Frequência:</b> Semestral  <b>Desloc. de Notificação:</b> 5  <b>Desloc. de Consulta:</b> 0	<b>Consulta:</b> <pre> SELECT s.name AS name,        e.enrollment_number AS enrollment_number,        SUM(c.credits) AS total_credits FROM students s, enrollments e, `class_enrollments` ce,        courses c, course_types ct, course_classes cc WHERE s.id = e.student_id AND e.id NOT IN (SELECT enrollment_id FROM dismissals) AND e.enrollment_status_id = 2 AND e.level_id=4 AND ce.enrollment_id = e.id AND ct.has_score = 1 AND ct.id = c.course_type_id /* Tópicos */ AND ct.id = 3 AND c.id = cc.course_id AND cc.id = ce.course_class_id GROUP BY s.name, e.enrollment_number HAVING SUM(c.credits) &gt; 8 ORDER BY s.name </pre>
	<b>(Um e-mail com todos os resultados)</b>
	<b>Destinatário:</b> <%= User.where(:role_id => 2).map(&:email).join(';') %>
	<b>Assunto:</b> Alunos de doutorado cursando mais de 8 créditos em disciplinas de Tópicos
	<b>Corpo:</b> Prezado Coordenador,  Informamos que os seguintes alunos de doutorado cursaram ou estão inscritos em mais de 8 créditos de disciplinas de Tópicos Especiais:  <pre> &lt;% records.each do  record  %&gt;   - &lt;%= record['name'] %&gt; (&lt;%= record['enrollment_number'] %&gt;), total de   créditos: &lt;%= record['total_credits'] %&gt; &lt;% end %&gt; </pre> Importante ressaltar que essa regra começou a ser checada automaticamente em Março de 2014.

21 – CORD: Alunos de doutorado cursando mais de uma disciplina de tópicos com seu orientador	
<b>Frequência:</b> Semestral  <b>Desloc. de Notificação:</b> 5  <b>Desloc. de Consulta:</b> 0	<b>Consulta:</b> <pre> SELECT s.name AS name,        e.enrollment_number AS enrollment_number,        SUM(c.credits) AS total_credits FROM students s, enrollments e, `class_enrollments` ce,        courses c, course_types ct, course_classes cc WHERE s.id = e.student_id AND e.id NOT IN (SELECT enrollment_id FROM dismissals) AND e.enrollment_status_id = 2 AND e.level_id=4 AND ce.enrollment_id = e.id AND ct.has_score = 1 AND ct.id = c.course_type_id /* Tópicos */ AND ct.id = 3 AND c.id = cc.course_id AND cc.id = ce.course_class_id /* Professor é o orientador */ AND cc.professor_id IN (SELECT professor_id FROM advisements WHERE enrollment_id = e.id AND main_advisor = 1) GROUP BY s.name, e.enrollment_number HAVING SUM(c.credits) &gt; 4 ORDER BY s.name </pre>
	<b>(Um e-mail com todos os resultados)</b>
	<b>Destinatário:</b> <%= User.where(:role_id => 2).map(&:email).join(';') %>
	<b>Assunto:</b> Alunos de doutorado cursando mais de 2 disciplinas de tópicos com seu orientador principal
	<b>Corpo:</b> Prezado Coordenador,  Informamos que os seguintes alunos de doutorado cursaram ou estão inscritos em mais de 4 créditos de disciplinas de Tópicos Especiais com seu orientador principal:  <pre> &lt;% records.each do  record  %&gt;   - &lt;%= record['name'] %&gt; (&lt;%= record['enrollment_number'] %&gt;), total de   créditos: &lt;%= record['total_credits'] %&gt; &lt;% end %&gt; </pre> Importante ressaltar que essa regra começou a ser checada automaticamente em Março de 2014.

22 – CORD: Bolsistas CAPES DS de doutorado que não cursaram 2 Estágio em Docência e não estão inscritos esse período	
<b>Frequência:</b> Semestral  <b>Desloc. de Notificação:</b> 16  <b>Desloc. de Consulta:</b> 0	<b>Consulta:</b> SELECT s.name AS name, e.enrollment_number AS enrollment_number, s.email AS email FROM students s, enrollments e WHERE s.id = e.student_id AND e.id NOT IN (SELECT enrollment_id FROM dismissals) AND e.enrollment_status_id = 2 /* Aluno regular */ AND e.level_id = 4 /* Aluno de doutorado */ AND e.id IN (SELECT sd.enrollment_id FROM scholarship_durations sd, scholarships s WHERE sd.scholarship_id = s.id AND s.sponsor_id = 10 AND s.scholarship_type_id = 2) /* Não cursou dois Estágio em Docência */ AND e.id NOT IN (SELECT en.id FROM enrollments en, class_enrollments ce, courses c, course_types ct, course_classes cc WHERE ct.id = c.course_type_id AND c.id = cc.course_id AND cc.id = ce.course_class_id AND en.id = ce.enrollment_id AND (ce.situation = "Aprovado" OR ce.situation = "Incompleto") AND ct.id = 7 GROUP BY enrollment_number HAVING count(*) >= 2) /* Não está inscrito esse período */ AND e.id NOT IN (SELECT en.id FROM enrollments en, class_enrollments ce, courses c, course_types ct, course_classes cc WHERE ct.id = c.course_type_id AND c.id = cc.course_id AND cc.id = ce.course_class_id AND en.id = ce.enrollment_id AND ce.situation = "Incompleto" AND ct.id = 7 ) /* Bolsista CAPES */ ORDER BY s.name
	<b>(Um e-mail com todos os resultados)</b>
	<b>Destinatário:</b> <%= User.where(:role_id => 2).map(&:email).join(';') %>
	<b>Assunto:</b> Bolsistas CAPES DS que precisam cursar Estágio em Docência
	<b>Corpo:</b> Prezado Coordenador,  Informamos que os seguintes bolsistas CAPES DS de doutorado não cursaram dois Estágio em Docência, e não estão inscritos esse período:  <% records.each do  record  %> - <%= record['name'] %> (<%= record['enrollment_number'] %>), email: <%= record['email'] %> <% end %>  Importante ressaltar que essa regra começou a ser checada automaticamente em Março de 2014.

23 – CORD: Bolsas não alocadas	
<p><b>Frequência:</b> Mensal</p> <p><b>Desloc. de Notificação:</b> 2</p> <p><b>Desloc. de Consulta:</b> 0</p>	<p><b>Consulta:</b> /* Bolsas ativas que terminam antes da data desejada */ SELECT s.scholarship_number, sd.cancel_date AS "data_termino", s.end_date AS "validade_da_bolsa" FROM scholarships s, scholarship_durations sd WHERE s.id = sd.scholarship_id AND ( s.end_date IS NULL OR s.end_date &gt; %{query_date} ) AND sd.end_date &lt; %{query_date} AND sd.cancel_date IS NULL UNION SELECT scholarship_number, "nao alocada" AS cancel_date, end_date AS "validade_da_bolsa" FROM scholarships s WHERE s.id NOT IN ( SELECT scholarship_id FROM scholarship_durations WHERE cancel_date IS NULL ) AND (s.end_date IS NULL OR s.end_date &gt; %{query_date}) UNION SELECT scholarship_number, "alocada a aluno desligado" AS cancel_date, end_date AS "validade_da_bolsa" FROM scholarships s WHERE s.id IN ( SELECT scholarship_id FROM scholarship_durations WHERE cancel_date IS NULL AND enrollment_id IN (SELECT enrollment_id FROM dismissals) )</p> <p><b>(Um e-mail com todos os resultados)</b></p> <p><b>Destinatário:</b> &lt;%= User.where(:role_id =&gt; 2).map(&amp;:email).join(';') %&gt;; secretaria.pos@ic.uff.br</p> <p><b>Assunto:</b> Bolsas disponíveis</p> <p><b>Corpo:</b> Prezado Coordenador</p> <p>Abaixo estão listados as bolsas que estão disponíveis atualmente:</p> <p>&lt;% records.each do  record  %&gt; - &lt;%= record['scholarship_number'] %&gt;, data de cancelamento = &lt;%= record['cancel_date'] %&gt;, validade da bolsa: &lt;%= record['validade_da_bolsa'] %&gt; &lt;% end %&gt;</p>