UNIVERSIDADE DE CAXIAS DO SUL
Centro de computação e Tecnologia da Informação
Curso de Bacharelado em Sistemas de Informação

JOVANI DALZOCHIO

IMPLEMENTAÇÃO DE NOVAS FUNCIONALIDADES PARA O PORTAL INTERGENICDB E POVOAMENTO DO SEU BANCO DE DADOS

CAXIAS DO SUL
2014

UNIVERSIDADE DE CAXIAS DO SUL
Centro de computação e Tecnologia da Informação
Curso de Bacharelado em Sistemas de Informação

JOVANI DALZOCHIO

IMPLEMENTAÇÃO DE NOVAS FUNCIONALIDADES PARA O PORTAL INTERGENICDB E POVOAMENTO DO SEU BANCO DE DADOS

Trabalho de Conclusão de Curso do Título de Bacharel pela Universidade de Caxias do Sul.
Área de concentração: Biologia Molecular
Orientador Prof Dr. Daniel Luis Notari

CAXIAS DO SUL
2014

RESUMO

Os recursos hoje disponíveis na área da informática e a necessidade de se trabalhar com os dados gerados por pesquisas na área de biologia molecular, como o projeto genoma humano, tornaram possível o surgimento de uma nova ciência chamada Bioinformática. Muitas ferramentas já foram criadas nessa área, portais que acessam informações geradas pelas pesquisas realizadas na área da biologia molecular são exemplos dessas ferramentas. O grupo de Bioinformática da Universidade de Caxias do Sul sentiu a necessidade de criar um banco de dados, denominado PromotoresDB, para armazenar sequências de nucleotídeos de regiões promotoras de organismos procariontes, já que os portais atualmente disponíveis não atendiam a esta demanda. Posteriormente, foi criado o Portal IntegenicDB para realizar consultas usando o banco de dados PromotoresDB. Contudo, é necessário que o portal seja verificado e validado quanto as suas funcionalidades, além de necessitar de uma ferramenta que seja capaz de buscar dados de outros bancos de dados para carregá-los constantemente. A partir desses requisitos, foi feita a modelagem e a geração dos artefatos necessários para o desenvolvimento do projeto. Depois de concluído o desenvolvimento da nova ferramenta e implementado os novos requisitos do portal, foram realizados testes utilizando as consultas do IntergenicDB para validar os dados importados. Com a conclusão do projeto, o portal IntergenicDB possuirá um ambiente mais ágil para navegação e população do banco de dados.

Palavras-Chave: IntergenicDB, Biologia Molecular, Bioinformática, Regiões Intergênicas, Promotores, Banco de Dados de Biologia Molecular.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

BDBM	Banco de Dados de Biologia Molecular (Molecular Biology Database)
CMR	Comprehensive Microbial Resource
DDBJ	DNA DataBank of Japan
DNA	Ácido Desoxirribonucleico (Deoxyribonucleic Acid)
EMBL	European Molecular Biology Laboratory
FTP	Protocolo de Transferência de Arquivos (File Transfer Protocol)
GEO	Gene Expression Omnibus
mRNA	RNA mensageiro (Messenger RNA)
NCBI	National Center for Biotechnology Information
NLM	National Library of Medicine
RNA	Ácido Ribonucleico (Ribonucleic Acid)
rRNA	RNA ribossômico (Ribosomal RNA)
SCV	Sistema de controle de versão
TDD	Desenvolvimento Guiado por Teste
tRNA	RNA de transferência (Transfer RNA)
UCS	Universidade de Caxias do Sul

SUMÁRIO

1 INTRODUÇÃO

A biologia molecular é a área da ciência pertencente à biologia que estuda a estrutura celular e os seus constituintes moleculares com suas interações e localização. Apesar de existirem similaridades entre as células, esses organismos mantêm diferenças fundamentais, sendo organizados em dois grandes grupos: os procariotos, que são sempre unicelulares; e os eucarióticos, que em sua maioria são constituídos de seres multicelulares, com alguns organismos unicelulares (ZAHA, 2001).

Tanto em um procarioto quanto em um eucarioto, o genoma tem a função de repositório replicativo da informação codificada pelo DNA (ácido desoxirribonucleico) que a constituiu (ZAHA, 2001). O genoma é composto por duas regiões principais: as gênicas, normalmente codificadoras de proteína; e as intergênicas, chamadas de regiões não-codificadoras (LESSA, 2012).

Um gene é uma sequência de nucleotídeos do DNA que é expresso em um produto funcional. Assim, o DNA é a molécula formadora dos genes (ALBERTS *et al.*, 1997; JUNQUEIRA *et al.*, 2005). DIAS CORREIA (2007) define RNA não codificante (ncRNA) como " o RNA que não codifica proteínas, embora este ncRNA contenha na sua estrutura informações variadas podendo pois ter funções especiais". OS ncRNAs eram chamados de "junk DNA" por se acreditar que não possuíam funções importantes. Descobriu-se ao longo dos anos exatamente o oposto, os ncRNA são importantes para as mais diversas funções do mecanismo celulares (LESSA, 2012).

A biologia tradicionalmente é uma ciência de observação. Nos últimos anos os dados biológicos se tornaram muito mais discretos. Grande parte da atividade organizada da bioinformática procura analisar os dados de um organismo. O fluxo e o controle dessas informações são o estado da arte da bioinformática. Com o avanço da computação e a o aumento da capacidade de armazenamento e processamento de dados na área de hardware e software a bioinformática se tornou possível (LESK, 2005).

Com a necessidade de se armazenar os dados biológicos, de processá-los e consultá-los de forma simples e eficiente, surgiram os BDBM (bancos de dados de biologia molecular), principalmente com o surgimento de sequenciadores

automáticos de DNA a partir da segunda metade da década de 90. Nos BDBMs são armazenados dados biológicos como sequências de nucleotídeos no GenBank[1] (GenBank Sequence Database), sequências de aminoácidos no PIR[2] (Protein Identification Resource), estruturas terciárias da proteína no PDB[3] (Protein Data Bank), entre outras (SEIBEL et al, 2000; CASTRAVECHI, 2004).

Além dos dados biológicos citados anteriormente, os BDBM podem armazenar informações de sequências de DNA denominadas promotores. Essas sequências sinalizam onde a síntese do RNA deve ser iniciada (ZAHA, 2001; JUNQUEIRA, 2005). Utilizando métodos computacionais de predição, é possível avaliar as chances de uma região ser ou não promotora (SEIBEL et al, 2000).

1.1 O PROBLEMA

O portal IntergenicDB é um portal de consultas por regiões promotoras (DAVANZO, 2010; PICOLLOTO, 2012). Sua criação aconteceu através de uma demanda do grupo de bioinformática da UCS para a criação de uma ferramenta para pesquisa e armazenamento de informações sobre Biologia Molecular (PICOLLOTO, 2012).

As informações acessadas pelo portal estão armazenadas no banco de dados PromotoresDB, que foi criado com o objetivo de armazenar e estudar as sequências intergênicas com possível função biológica de DNA de organismos procariontes (DAVANZO, 2010).

Através do portal é possível fazer buscas usando os dados de regiões intergênicas. Estas buscas podem ser refinadas usando-se como parâmetros de pesquisa o gene, fatia de região intergênica e por organismo. O resultado da pesquisa pode ser salvo. Além disto, o usuário do portal poderá submeter arquivos com informações a serem armazenadas no banco de dados (DAVANZO, 2010).

É preciso fazer modificações no IntergenicDB, visando aperfeiçoá-lo e melhorá-lo, tais como:

- Melhorias na forma de consulta e exibição dos métodos de predição;

[1] http://www.ncbi.nlm.nih.gov/Genbank/index.html
[2] http://www-nbrf.georgetown.edu/
[3] http://www.rcsb.org/pdb

- Oferecer uma alternativa para que os usuários cadastrados possam recuperar sua senha;
- Garantir que as consultas atualmente disponíveis no portal estejam funcionando;
- Realizar testes de busca e validações, garantido que os dados retornados pelo IntergenicDB estejam de acordo com os filtros usados na busca;
- Testar as funcionalidade do portal e sanar qualquer inconformidade encontrada.
- Popular o banco de dados PromotoresDB utilizando os dados já existentes em outros portais de biologia molecular. Os portais que serão utilizados para a busca de informações serão o NCBI[4] (National Center for Biotechnology Information) e o JCVI – CRM[5] (The Comprehensive Microbial Resource) (PICOLLOTO, 2012).

1.2 QUESTÃO DE PESQUISA

Baseado no problema de pesquisa descrito anteriormente e dentro do contexto apresentado surge a questão de pesquisa:

Quais as melhorias que o portal IntergenicDB necessita e quais são as novas ferramentas que precisarão ser criadas para fazer do portal um ambiente mais acessível de usar e mais ágil para efetuar cargas de informação no banco de dados?

1.3 OBJETIVOS

O objetivo deste trabalho é desenvolver casos de teste para validar as funcionalidades do portal IntergenicDB, validá-los e fazer a carga dos novos dados para o banco de dados.

1.3.1. Objetivos Específicos

[4] http://www.ncbi.nlm.nih.gov/
[5] http://cmr.jcvi.org/tigr-scripts/CMR/CmrHomePage.cgi

Para alcançar o objetivo apresentado, o presente trabalho se orientará pelos objetivos específicos apresentados abaixo:

- Definir os requisitos e gerar os artefatos de software da nova ferramenta de importação de dados;
- Implementar a nova ferramenta de importação de dados;
- Definir e especificar as melhorias que deverão ser implementadas no portal IntergenicDB.
- Implementar as novas funcionalidades do portal, utilizando os padrões já empregados no IntergenicDB;
- Definir a metodologia que será utilizada para aplicação dos casos de teste;
- Criar casos de testes a serem aplicadas as consultas do IntergenicDB;
- Aplicar os casos de testes;
- Avaliar os resultados;

1.4 ESTRUTURA DO TEXTO

O capítulo 2 apresenta uma introdução à biologia molecular, conceituando a estrutura celular de organismos procariotos (DNA e RNA) e alguns de seus processos. Também são definidos conceitos de bioinformática, bancos de dados biológicos e os portais de bioinformática.

O capítulo 3 apresenta o portal IntergenicDB e o banco de dados PromotoresDB.

O capítulo 4 apresenta os novos requisitos que serão empregados no IntergenicDB ou na nova ferramenta de importação de dados.

No capítulo 5 é apresentado o desenvolvimento dos requisitos do portal e desenvolvimento da nova ferramenta de importação de dados propostos no capítulo 4. Também é apresentado o resultado dos testes efetuados na ferramenta de consulta do IntergenicDB com as informações obtidas através dos dados fornecidos pela nova ferramenta de importação

O capítulo 6 apresenta a conclusão do trabalho e sugestões para futuros trabalhos.

2 BIOLOGIA MOLECULAR

A biologia molecular é a área da ciência pertencente a biologia que estuda a estrutura celular e os seus constituintes moleculares com suas interações e localização (ZAHA, 2001). É tradicionalmente uma ciência de observação. Nos últimos anos os dados biológicos se tornaram muito mais discretos, como no caso do sequenciamento de nucleotídeos e aminoácidos. Grande parte da atividade organizada da bioinformática procura analisar os dados de um organismo e entender seus processos em nível moleculares. Com o avanço da computação e a o aumento da capacidade de armazenamento e processamento de dados na área de hardware e software a bioinformática se tornou viável, assim é possível fazer inferências a partir de um grande volume de dados de biologia molecular, fazer conexões e posteriormente derivar predições (LESK, 2005).

O objetivo deste capítulo é apresentar os componentes biológicos e as reações que ocorrem entre eles. Além disso, serão apresentados os bancos de dados de biologia molecular e os portais que fazem acesso a esses bancos de dados, tornando compreensíveis as próximas seções deste trabalho.

2.1 GENE, DNA E RNA

JUNQUEIRA E CARNEIRO (2005) define gene como “uma sequência de nucleotídeos do DNA que é expresso em um produto funcional”. Deste modo o gene é constituído por polímeros lineares de nucleotídeos, unidos por ligações fosfodiéster (ZAHA, 2001).

O DNA é uma molécula em forma de dupla hélice (Figura 1), sendo suas bases unidas por pontes de hidrogênio (JUNQUEIRA E CARNEIRO, 2005). Existem quatro tipos de bases, a adenina (A), a citosina (C), a guanina (G) e a timina (T). As pontes de hidrogênio se formam em número maior entra pares de bases compostos por G e C ou entre A e T. A função do DNA é a de comandar todo o funcionamento da célula e de transferir informação genética para as células filhas (ALBERTS et al., 1997).

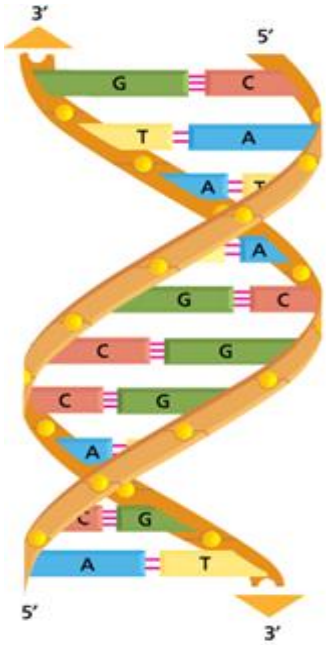

Figura 1: Estrutura do DNA

Fonte: (ALBERTS et al., 2011).

Diferente de um DNA que é formado por duas cadeias de polinucleotídeos, o Ácido Ribonucleico (RNA) é uma molécula formada geralmente por um filamento único (Figura 2). A fita de RNA simples é criada através da transcrição de DNA, com o RNA retendo toda informação da sequência de DNA ao qual foi copiado (ALBERTS et al., 1997).

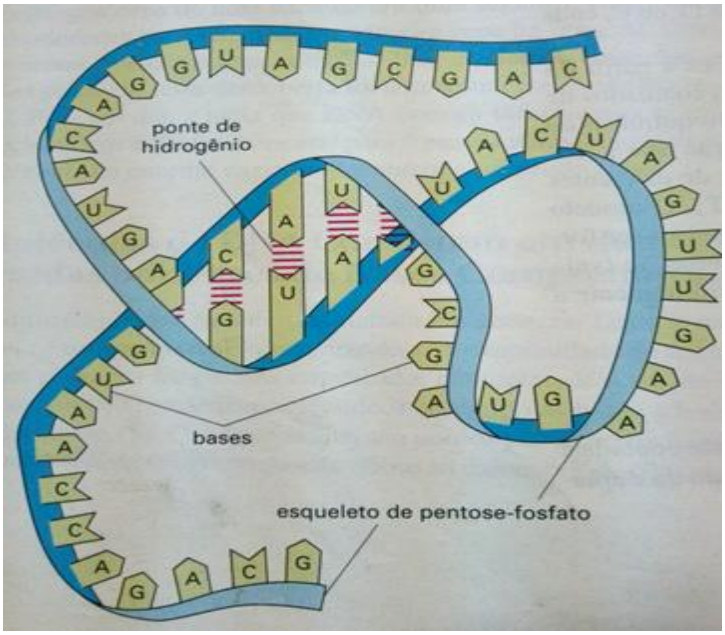

Figura 2: Estrutura do RNA

Fonte: (ALBERTS et al., 1997)

JUNQUEIRA E CARNEIRO (2005) distingue o RNA em três variedades principais do ponto de vista estrutural funcional:

- RNA mensageiro (mRNA), que tem a função de determinar a posição dos aminoácidos nas proteínas;
- RNA de transferência (tRNA). Sua função é transportar os aminoácidos unindo o seu anticódon ao códon do mRNA e determinar a posição dos aminoácidos nas proteínas;
- RNA ribossômico (rRNA), que combinado com o mRNA forma os polirribossomos;

A região intergênica é a região do DNA que não possui informações para a síntese do RNA. É dentro da região intergênica que está localizada a sequência de DNA denomina promotor. A função do promotor é a de indicar o local que indica a síntese de RNA deve ser iniciada (Figura 3). As moléculas de RNA são sintetizadas por enzimas denominadas RNA polimerase, que se liga fortemente a região promotora (ALBERTS et al., 1997).

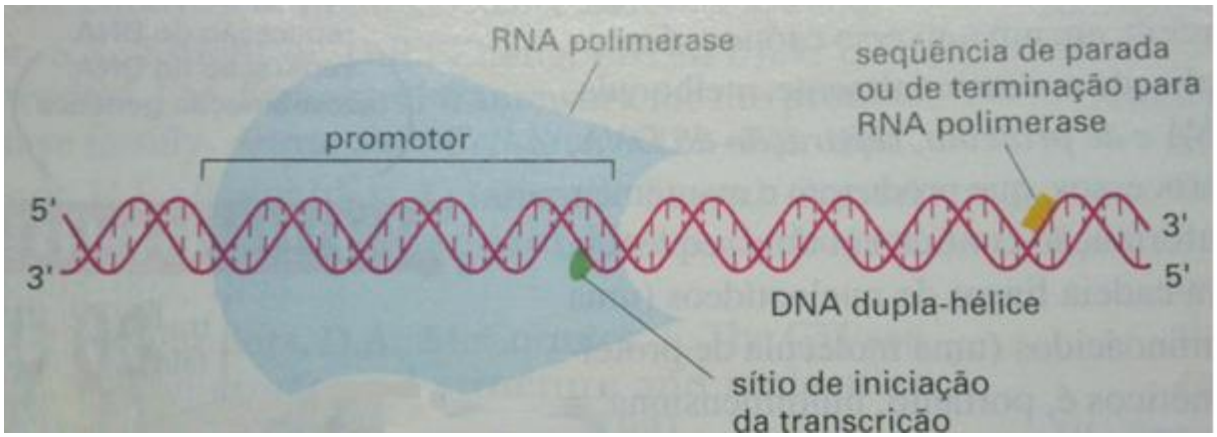

Figura 3 - Região promotora

Fonte: (ALBERTS et al., 1997)

No processo de transcrição (Figura 4) uma das fitas de DNA atua como um molde para o pareamento das bases.

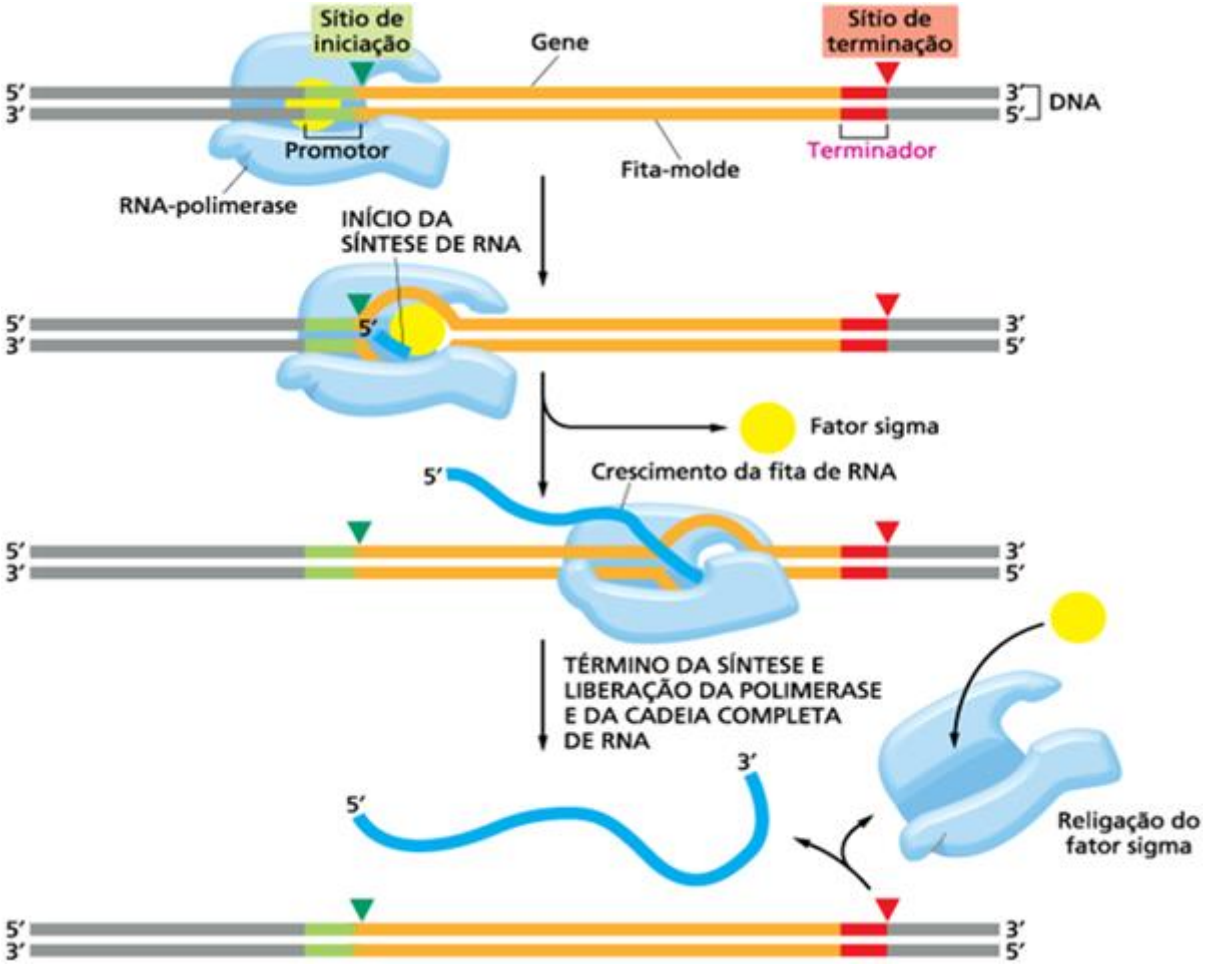

Figura 4 - Processo de síntese do RNA

Fonte: (ALBERTS et al., 2011).

A molécula de RNA polimerase se desloca pausadamente através da fita de DNA exposta. O processo continua até local onde está localizada uma cadeia de DNA especial, que sinaliza a parada da transição. Ao final o RNA polimerase libera as duas fitas de DNA e a recém-sintetizada cadeia de RNA (ALBERTS et al., 1997).

No processo de replicação do DNA, tantos as células eucariontes como as células procariontes possuem enzimas que desempenham esse papel, tais enzimas são denominadas DNA-Polimerase. A síntese de uma nova cadeia complementar é obtida através do desenrolamento da cadeia de dupla-hélice. Ao se abrir a dupla-hélice, duas novas cadeias se formam, sendo uma em cada hélice, originando assim duas novas moléculas com pode ser visto na Figura 5. Esse processo de replicação possui a característica de ser semiconservativo, pois cada nova molécula é uma cópia perfeita de uma molécula preexistente (JUNQUEIRA E CARNEIRO, 2005).

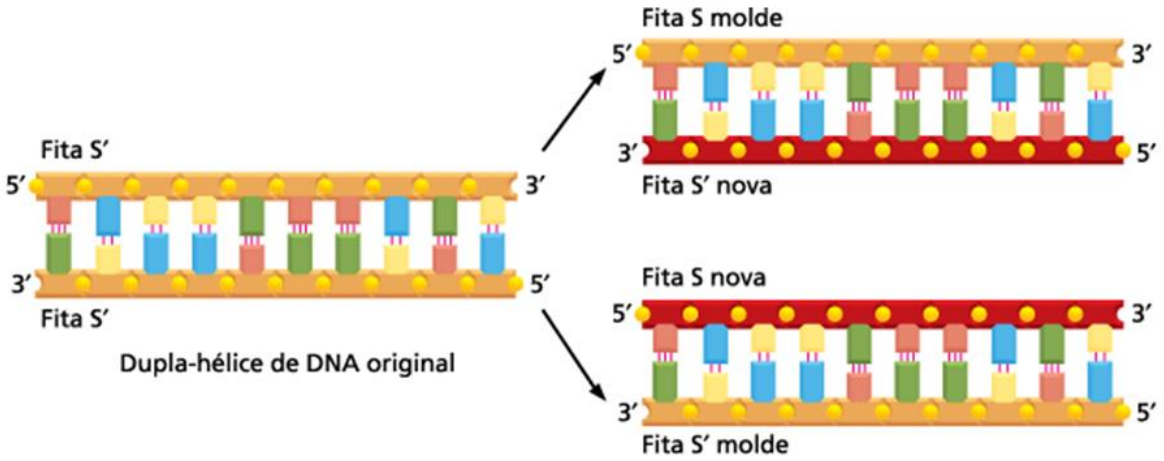

Figura 5 - Processo de Replicação do DNA

Fonte: (ALBERTS et al., 2011).

2.2 BANCO DE DADOS DE BIOLOGIA MOLECULAR

O genoma tem a função de repositório de toda informação genética, estando codificada pelo ácido desoxirribonucleico (DNA) que compõe esse genoma (ZAHA, 2001). O projeto genoma humano, que teve como propósito sequenciar e mapear todos os genes da espécie humana teve suas informações armazenadas em BDBMs (NOTARI, 2012).

Segundo LESK (2008) o proteoma, em analogia ao genoma, “é o conjunto de proteínas de um organismo”. O estudo desse conjunto de proteínas busca identificar, localizar e analisar suas funções. Essa é uma área de estudos que trabalha com um grande volume de informações que precisam de coleta e análise de dados em larga escala (LESK, 2008).

Transcriptoma refere-se ao completo de transcritos presentes em uma célula. Nesse caso, os BDBMs são utilizados, por exemplo, com o objetivo de coletar e classificar diferentes tipos de RNA encontrados em organismos. Um exemplo de banco de dados de transcriptoma é o GEO (Gene Expression Omnibus), que armazena dados sobre expressões de gene e arranjos de hibridização (NOTARI, 2012).

Com a necessidade de se armazenar os dados biológicos citados acima, de processá-los e consultá-los de forma simples e eficiente, os dados biológicos foram organizados em bancos de dados de biologia molecular (BDBM). Isto ocorreu

fortemente com o surgimento de sequenciadores automático de DNA a partir da segunda metade da década de 90 (SEIBEL et al, 2000; CASTRAVECHI, 2004).

Nos BDBMs são armazenados dados biológicos como sequências de nucleotídeos no GenBank[6] (GenBank Sequence Database), sequências de aminoácidos no PIR[7] (Protein Identification Resource), estruturas terciárias da proteína no PDB[8] (Protein Data Bank), entre outras (SEIBEL et al, 2000; CASTRAVECHI, 2004).

Um banco de dados é um arquivo de informações com uma organização estruturada das informações e deve ter ferramentas para seu acesso (LESK, 2005). Os BDBMs contêm sequências de ácidos nucléicos e de proteínas, estruturas e funções de macromoléculas, padrões de expressão, redes de vias metabólicas e cascatas de regulação.

As próximas seções descrevem o portal NCBI[9] (National Center for Biotechnology Information) e o portal Comprehensive Microbial Resource[10] (CMR). Estes portais fornecem acessos a BDBMs, os quais servirão de fonte de dados para o portal IntergenicDB.

2.2.1. NCBI

O NCBI foi criado em 1988 como uma divisão do NLM (National Library of Medicine), por iniciativa do senador Claude Pepper (SMITH, 2013). A missão do NCBI é desenvolver novas tecnologias de informação para ajudar na compreensão dos processos moleculares e genéticos que controlam a saúde e as doenças (SMITH, 2013). O NCBI tem a tarefa de criar sistemas para armazenar e processar informações sobre biologia molécula, bioquímica e genética, facilitando o acesso a esses dados pela comunidade médica (SMITH, 2013).

O NCBI mantem e disponibiliza vários bancos de dados para consulta. Pode-se citar o PubMed[11], uma ferramenta de busca composta[12] por mais de 22 milhões de citações para a literatura biomédica do MEDLINE, revistas de ciências da vida, e

[6] http://www.ncbi.nlm.nih.gov/Genbank/index.html
[7] http://www-nbrf.georgetown.edu/
[8] http://www.rcsb.org/pdb
[9] http://www.ncbi.nlm.nih.gov/
[10] http://cmr.jcvi.org/tigr-scripts/CMR/CmrHomePage.cgi
[11] http://www.ncbi.nlm.nih.gov/pubmed/
[12] Consulta realizada em 23 de outubro de 2013

livros on-line. Outro banco de dados que pode ser citado é o Genbank[13], sendo parte da International Nucleotide Sequence Database Collaboration (Colaboração Internacional de Banco de Dados de Sequências de Nucleotídeos) juntamente com outras duas organizações, o DDBJ (DNA DataBank of Japan) e o EMBL (European Molecular Biology Laboratory). Estas três entidades trocam dados entre si todos os dias. O Genbank possui[14] atualmente mais de 154 milhões de bases e mais de 167 milhões de sequencias (MIZRACHI, 2013).

No Genbank são armazenados sequências de nucleotídeos e proteínas, além de informações sobre cada sequência com o nome científico e a taxonomia do organismo de origem. Para acessar toda essa informação é utilizado o Entrez[15]. As consultas efetuadas no portal ao banco de dados Genbank utilizam o aplicativo Entrez como mecanismo de busca (SEIBEL et al, 2000).

É possível ter acesso aos dados dos bancos de dados do NCBI fazendo um acesso remoto utilizando um programa cliente, com isso é possível realizar consultas baseadas no Entrez. Utilizando programas oferecidos pela ferramenta E-Utilities[16] é possível ter acesso aos dados de um servidor utilizando uma sintaxe fixa de URL, com um conjunto de parâmetros pré-estabelecidos (NOTARI, 2012). Através da Figura 6 é possível visualizar os dados retornados de uma consulta no banco de dados de nucleotídeos do NCBI. Nessa consulta foi usada a URL "*http://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=nuccore&id=34577062,24475906&rettype=fasta&retmode=text".* Os parâmetros informados foram o banco de dados(db), a lista de Ids dos registros que devem ser retornados(&id), o tipo de arquivo utilizado (&rettype) e o formato dos dados (&retmode).

```
>gi|34577062|ref|NM_001126.2| Homo sapiens adenylosuccinate synthase
(ADSS), mRNA
GGAAGGGGCGTGGCCTCGGTCCGGGGTGGCGGCCGTTGCCGCCACCAGGGCCTCTTCCTGCGGGCGGTGC
TGCCGAGGCCGGCCTGCGCGGGGCAGTCATGGTACCCCCTTGAGCGGGCTGTGGCGGAGAGCGGGGCGGG
GACTGGCTGGAGGGTGGCGGCCCGGCGGGGCGGGGGCGGGGCCGGCCTCTGGCTCCTTCTTCCTCTGCAT
GTGGCTGGCGGCCGCAGAGCAGTTCAGTTCGCTCACTCCTCGCCGGCCGCCTCTCCTTCGGGCTCTCCTC
GCGTCACTGGAGCCATGGCGTTCGCCGAGACCTACCCGGCGGCATCCTCCCTGCCCAACGGCGATTGCGG
```

Figura 6 - Exemplo de dados do portal NCBI

[13] http://www.ncbi.nlm.nih.gov/genbank/
[14] Consulta realizada em 23 de outubro de 2013
[15] http://www.ncbi.nlm.nih.gov/gquery
[16] http://www.ncbi.nlm.nih.gov/books/NBK25500/

2.2.2. JCVI – CMR

O CMR é uma ferramenta que permite aos pesquisadores acesso a sequência de genomas de bactérias. O CMR instituiu que a atualização dos dados ocorre de forma trimestral. No início de cada trimestre os novos genomas disponíveis no GenBank são analisados e então são adicionados ao CMR.

O CMR oferece uma variedade de ferramentas e recursos que são disponibilizadas para acesso através da barra de menu existente no topo esquerdo do *site.* As ferramentas oferecidas pelo CMR são as seguintes (DAVIDSEN et al., 2009):

- *Genome Tools:* Busca de listas de organismos bem como um sumário de informações e análises por organismo selecionado;
- *Searches:* Permite realizar buscas por gene, genoma, regiões de sequência e evidências;
- *Comparative Tools:* Compara múltiplos genomas baseado em uma variedade de critérios, incluindo homologia e atributos genéticos.
- *Lists:* Permite listar informações sobre organismos como todos os genes de um genoma.
- *Downloads:* Permite o *download* de sequencias genéticas ou atributos por organismos. Permite também acessar o diretório FTP.
- *Carts:* Permite ao usuário selecionar genomas ou genes de preferência, ficando armazenado no *Genome Cart.* É possível fazer o *download* das preferências armazenadas no *Genome Cart;*

As ferramentas listadas acima fazem acesso ao Omniome (Figura 7), que é a base de dados relacional utilizada pelo CMR (DAVIDSEN et al., 2009). O banco de dados armazena informações de sequências de DNA, de proteínas, de genes e RNA entre outras características. Atualmente, o banco de dados possui[17] em sua base 672 bactérias, 48 *Archaea* (organismos parecidos com as bactérias) e 3 vírus.

[17] Consulta realizada em 23 de outubro de 2013

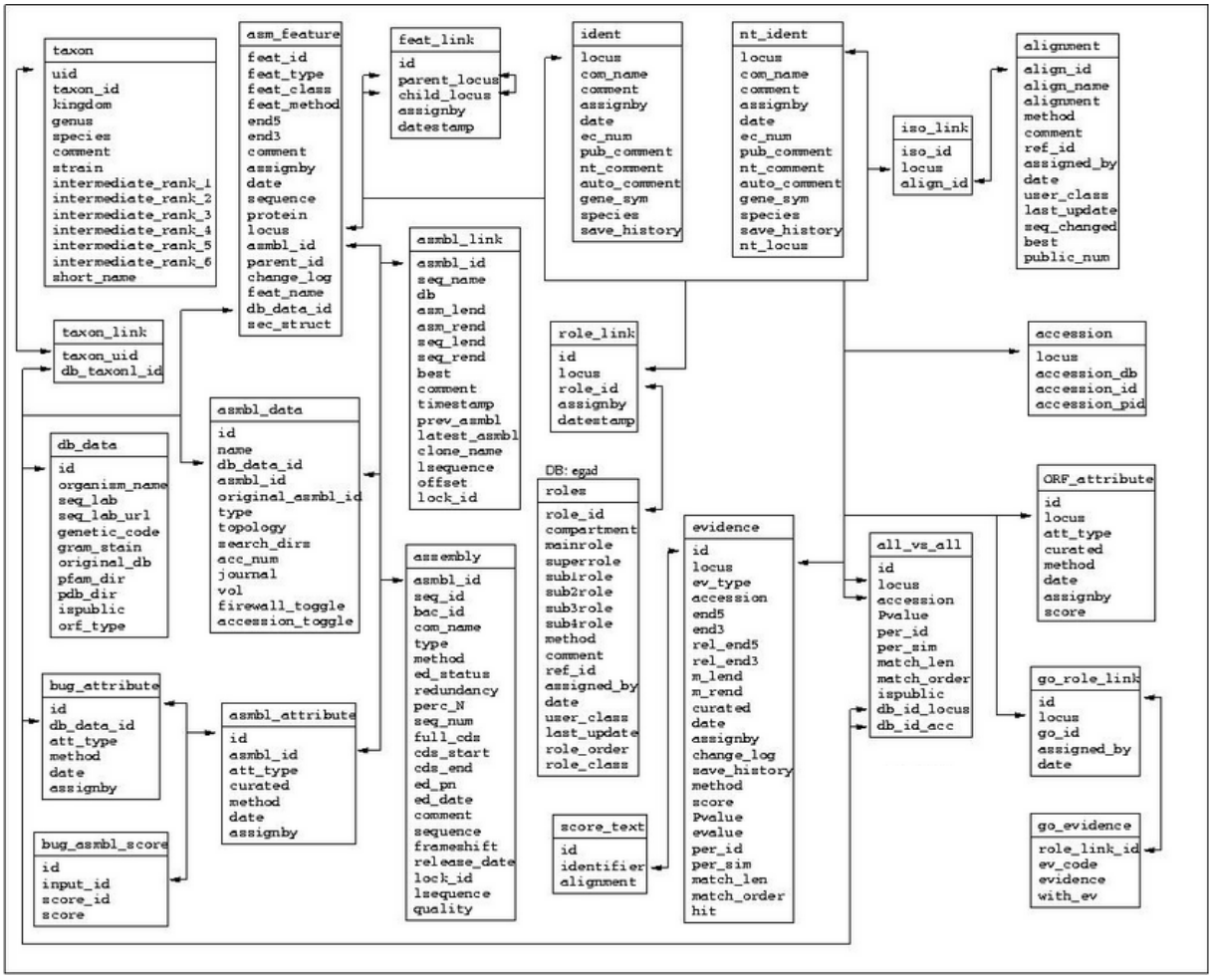

Figura 7 - Modelo lógico do banco de dados Omniome

Fonte: [18]

2.3 CONSIDERAÇÕES FINAIS

A biologia molecular é uma ciência que gera um grande volume de dados. Para poder armazenar todos esses dados biológicos são utilizados bancos de dados. O acesso a esses bancos de dados se dá através do uso de algumas ferramentas, como portais *web*. Essas ferramentas possibilitam que qualquer usuário possa fazer consultas e utilizar as informações armazenadas nesses bancos de dados para realizar pesquisas na área.

Como os bancos de dados até então existentes não contemplavam as necessidades de pesquisa sobre os promotores realizada pelo grupo de

[18] ftp://ftp.jcvi.org/pub/data/Omniome_Database/tab_delimited_omniome/

bioinformática da Universidade de Caxias do Sul, foi criado um novo banco de dados. Esse banco de dados denominado PromotoresDB foi proposto pelo aluno Aurione Francisco Molin (MOLIN, 2009). Posteriormente a aluna Vanessa Davanzo (DAVANZO, 2010) desenvolveu o portal IntergenicDB para que fosse possível interagir com o banco de dados. Em 2012, o aluno Douglas Picolotto (PICOLLOTO, 2012) implementou novas funcionalidades ao portal IntergenicDB, tornando-o mais robusto. No próximo capítulo será apresentado o banco de dados PromotoresDB e o portal IntergenicDB, para que seja possível compreender o seu funcionamento.

3 PORTAL INTERGENICDB

O IntergenicDB[19] é um portal que possui acesso a informações de sequências de DNA conhecidas como regiões intergênicas (ou promotores) de seres procariotos.

O portal surgiu como uma necessidade do grupo de Bioinformática da Universidade de Caxias do Sul (UCS) para armazenamento de informações sobre Biologia Molecular, já que os inúmeros Bancos de Dados de Biologia Molecular existentes não atendiam por completo as necessidades do grupo de Bioinformática.

O desenvolvimento do portal teve início no ano de 2009 pelo aluno Aurione Francisco Molin (MOLIN, 2009) com a modelagem e implementação do protótipo do banco de dados denominado “promotoresdb”. No ano de 2010 a aluna Vanessa Davanzo (DAVANZO, 2010) deu continuidade com a criação do portal IntergenicDB para que fosse possível interagir com o banco de dados. Em 2012, o aluno Douglas Picolotto (PICOLLOTO, 2012) implementou novas funcionalidades ao portal IntergenicDB, tornando o portal uma ferramenta mais robusta e com maior controle sobre as informações. Nas próximas seções serão apresentadas as características e o funcionamento do banco de dados PromotoresDB e do portal IntergenicDB.

3.1 PROMOTORESDB

O PromotoresDB surgiu da necessidade da professora e doutora Sheila de Ávila e Silva (Ávila e Silva, 2012) completar parte de seu trabalho para sua tese de doutorado intitulada “Redes neurais artificiais aplicadas no reconhecimento de regiões promotora sem bactérias Gram-negativas” (MOLIN, 2009).

O objetivo do banco de dados é o armazenamento de sequências de nucleotídeos de regiões promotoras de organismos procariontes. A partir desse objetivo, foram definidos os requisitos para o funcionamento do banco de dados, sendo eles (MOLIN, 2009):

- Regiões promotoras então ligadas a um organismo, este organismo possui um nome, um reino, uma família, e um tipo de molécula;

[19] http://www.danlian.com.br/intergenicdb/

- Os genes são possui um nome que para identificação, um símbolo, o número da posição de início (5'), o número da posição de fim (3'), uma função, uma quantidade de nucleotídeos "GC" na sequência do gene;
- Regiões Intergênicas caracterizam pelo número da posição de inicio da sequência (5'), pelo número da posição de fim da sequência (3'), pelo tamanho da sequência, pela sequência de nucleotídeos e o tipo de fita que pertence;
- Região Intergênica de teste possuem cadeias de 60 nucleotídeos que serão usados como fonte de dados usada para a predição que definirá se a sequência é candidata a ser promotora;
- Algoritmos denominados de métodos preditores são usados na predição das chances de uma região ser promotora, o método possui um nome que o identifica, o percentual de uma cadeia ser promotora e uma descrição das suas características de funcionamento ou aplicabilidade;
- Manter informações bibliográficas de publicações como informações do autor, título da obra, data da publicação, instituto ao qual está vinculado (exemplo – universidade), cidade e país ao qual foi produzido, um endereço web para acesso (caso esteja publicado na web), e um espaço para informações adicionais;

A ferramenta de banco de dados escolhida para o desenvolvimento do PromotoresDB foi o MySQL por ser o de código aberto e bastante popular no mundo, além de ser uma alternativa segura e não possuir nenhum custo para obtenção de licença de uso (MOLIN, 2009). O modelo ER que representa os dados do PromotoresDB juntamente com o portal IntergenicDB proposto pelo aluno Aurione Francisco Molin (MOLIN, 2009), juntamente com as alterações feitas pela aluna Vanessa Davanzo (DAVANZO, 2010) e pelo aluno Douglas Picolotto (PICOLLOTO, 2012) está representado na Figura 8.

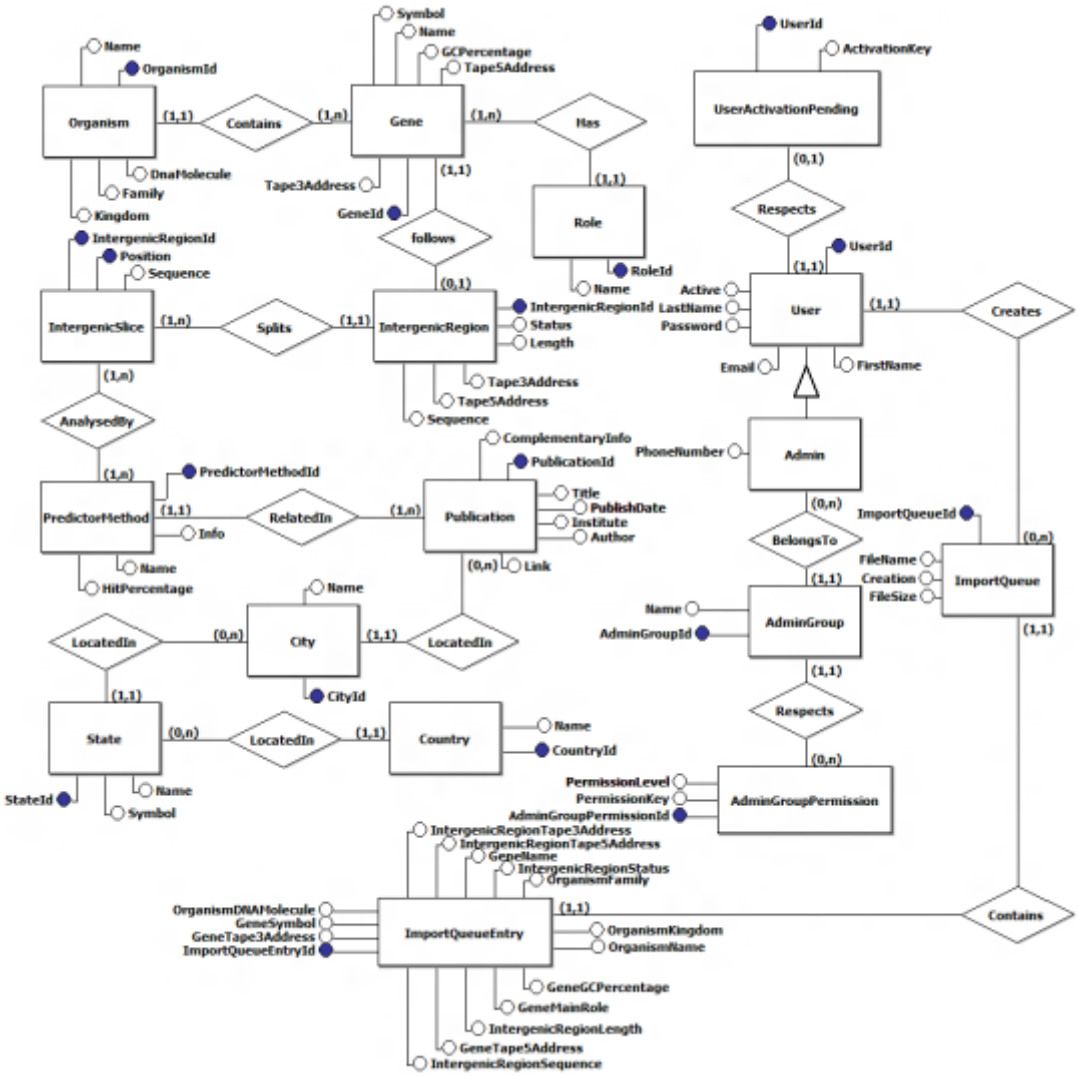

Figura 8 - Modelo conceitual do banco de dados PromotoresDB

Fonte: (PICOLLOTO, 2012)

3.2 INTERGENICDB

Para que fosse possível interagir com o PromotoresDB, verificou-se a necessidade de criação de uma nova ferramenta. Essa nova ferramenta deveria ser capaz de fazer consultas ao banco de dados, além de realizar *upload* e *download* de arquivos. Com isto foi criado o portal IntergenicDB (Figura 9).

Figura 9 - Portal IntergenicDB

Fonte: [20]

Para implementar o portal IntergenicDB, a ferramenta escolhida foi o ASP.NET, utilizando a linguagem C#. O portal utiliza uma máquina servidora *web* com um SGBD *MySQL* (Figura 10) que recebe as requisições de serviço e as encaminha para o *Controller*, que consulta o *Model* (modelo de dados) e retorna as informações do banco de dados para a *View*, onde o usuário irá interagir (DAVANZO, 2010). Já na modelagem, foram aplicados alguns padrões de projetos largamente utilizados como:

- O Model-View-Controller. Consiste em três objetos. O *Model,* que representa informações sobre o domínio de negócio, o *View,*, que é a representação do *Model* na tela e o *Controller*, controla as interações do usuário com a *View* e o modelo de dados (FOWLER et al., 2002);
- Abstract Factory: Possibilita a criação de um interface para um família de objetos sem especificar suas classes concretas (GAMMA et al.,1994);
- Data Gateway: Centraliza as operações de acesso aos dados em seu interior (FOWLER et al., 2002).

[20] http://www.danlian.com.br/intergenicdb/

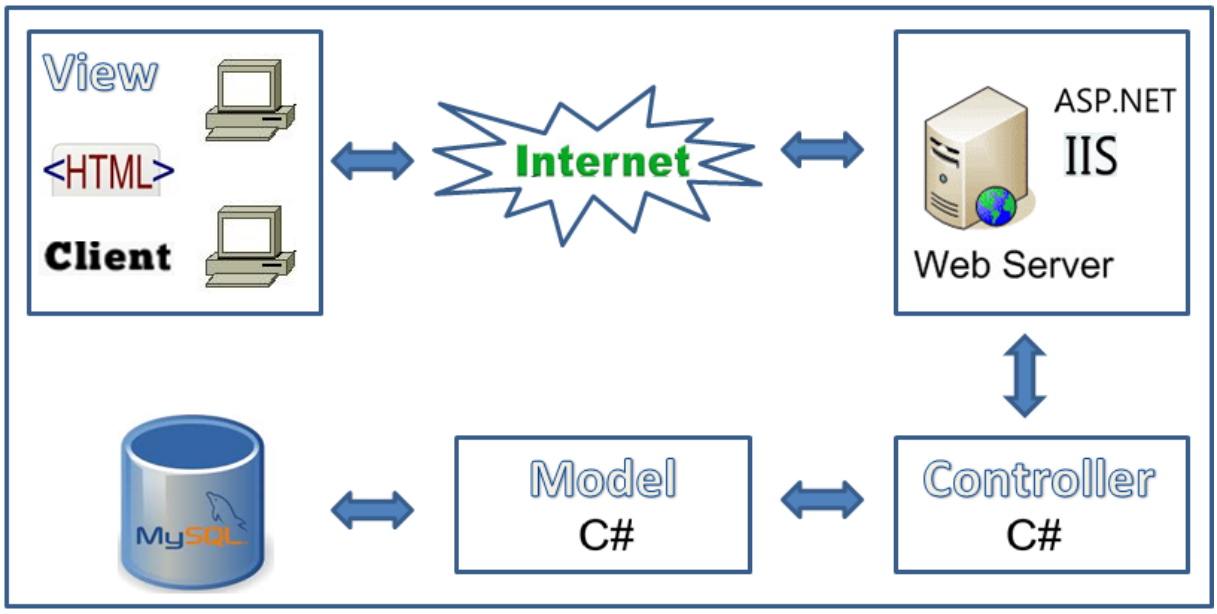

Figura 10 - Arquitetura do sistema

Fonte: (DAVANZO, 2010)

No ano de 2012 o aluno Douglas Picolloto implementou algumas novas funcionalidades ao portal IntergenicDB. Entre as melhorias empregadas pelo aluno Douglas Picolloto (PICOLLOTO, 2012), pode-se citar a nova área de administração criada para permitir a manutenção dos dados (Figura 12). Essa nova área foi dividida em três grandes partes. O grupo *Access Control* (Controle de Acesso) permite dar manutenção nas permissões e autenticações dos usuários. O grupo *Molecular Biology Information* (Informações sobre Biologia Molecular) permite a manutenção dos cadastros utilizados nas consultas dos sistemas. O grupo *Other* (Outros) dá ao usuário a possibilidade de gerenciar artigos e publicações postados pelos usuários do sistema, além de dar acesso a fila de arquivos para importação que foram submetidos pelos usuários, dando a opção de confirmar a importação dos dados ou de descartá-los (PICOLOTTO, 2012).

Os requisitos que deram origem a essas novas funcionalidades, juntamente com as funcionalidades implementadas pela aluna Vanessa Davanzo podem ser vistos através da Figura 11.

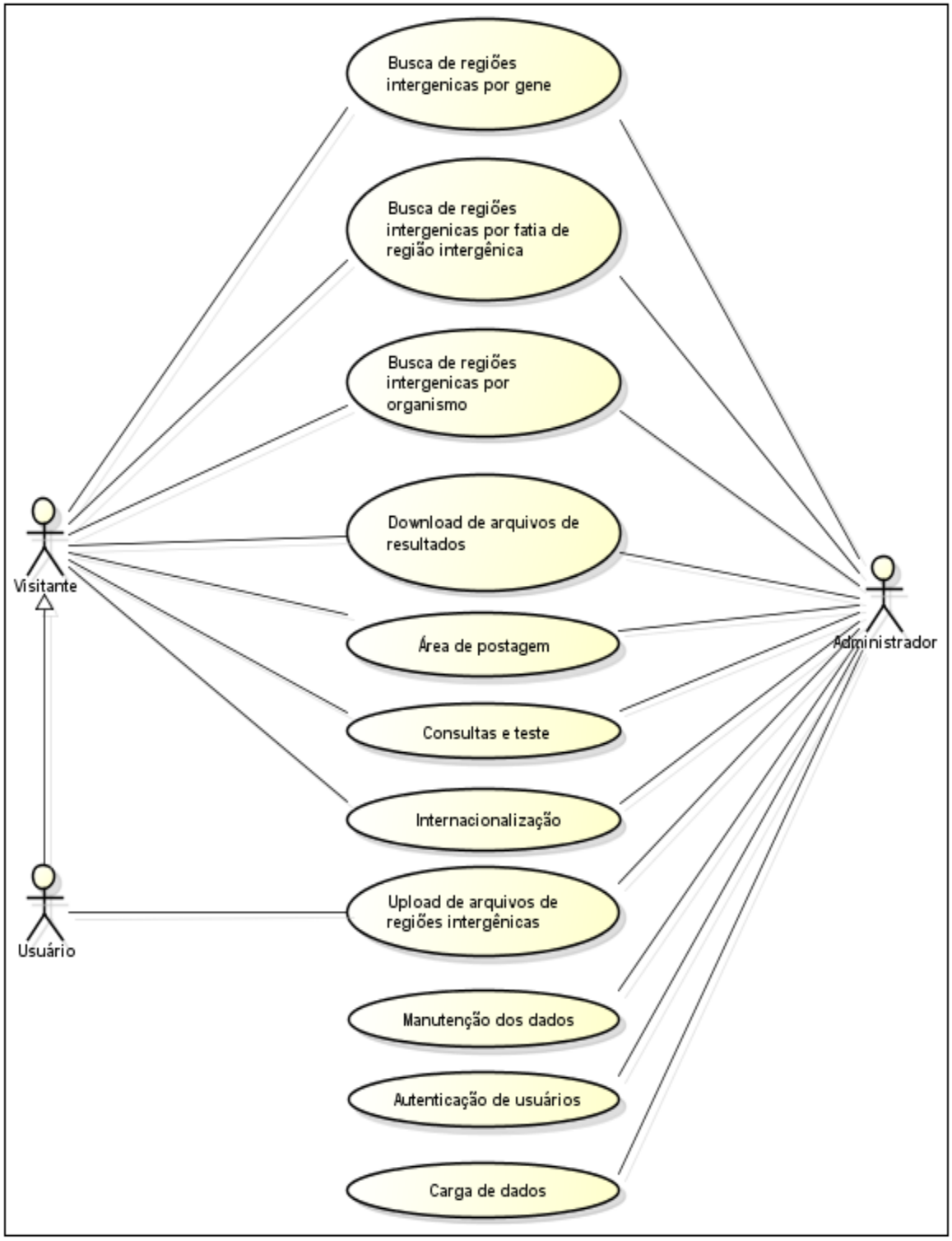

Figura 11 - Casos de uso das novas funcionalidades

Fonte: (DAVANZO, 2010; PICOLOTTO, 2012)

Essa nova área de administração (Figura 12) esta disponível através do endereço do portal, acrescentando "/Admin"[21] no final do endereço (PICOLOTTO, 2012).

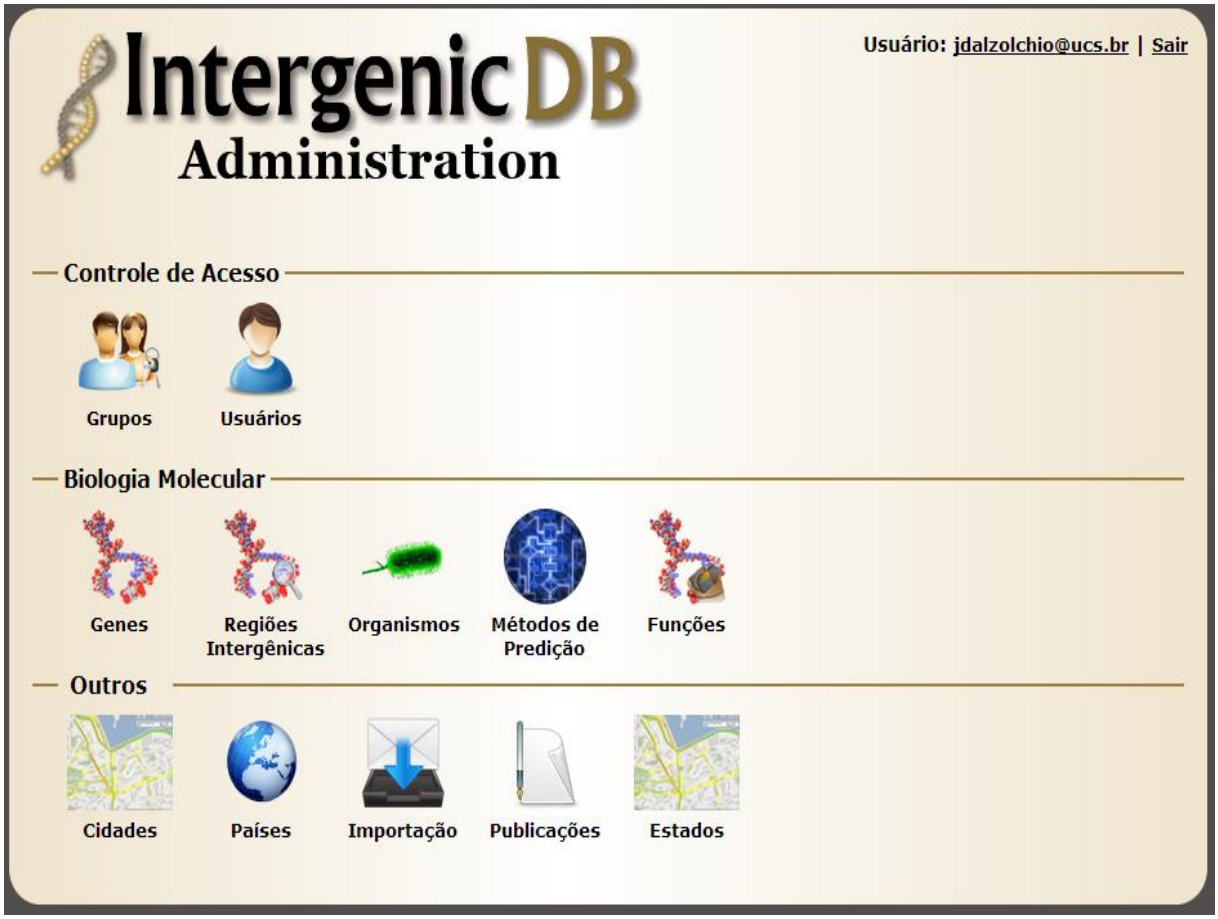

Figura 12 - Área de administração do portal IntergenicDB

Fonte: [22]

Atualmente o IntergenicDB já conta com algumas informações cadastradas pelo aluno Douglas Picolotto (PICOLOTTO, 2012). Essas informações foram fornecidas pela Profa. Dra. Scheila de Ávila e Silva em arquivo texto para importação pelo portal. Uma amostra de como os dados atualmente disponíveis são exibidos pelo portal para o usuário através de uma consulta poder ser vista através da Figura 13.

[21] Exemplo: http://www.intergenicdb.com.br/Admin
[22] http://www.danlian.com.br/intergenicdb/Admin

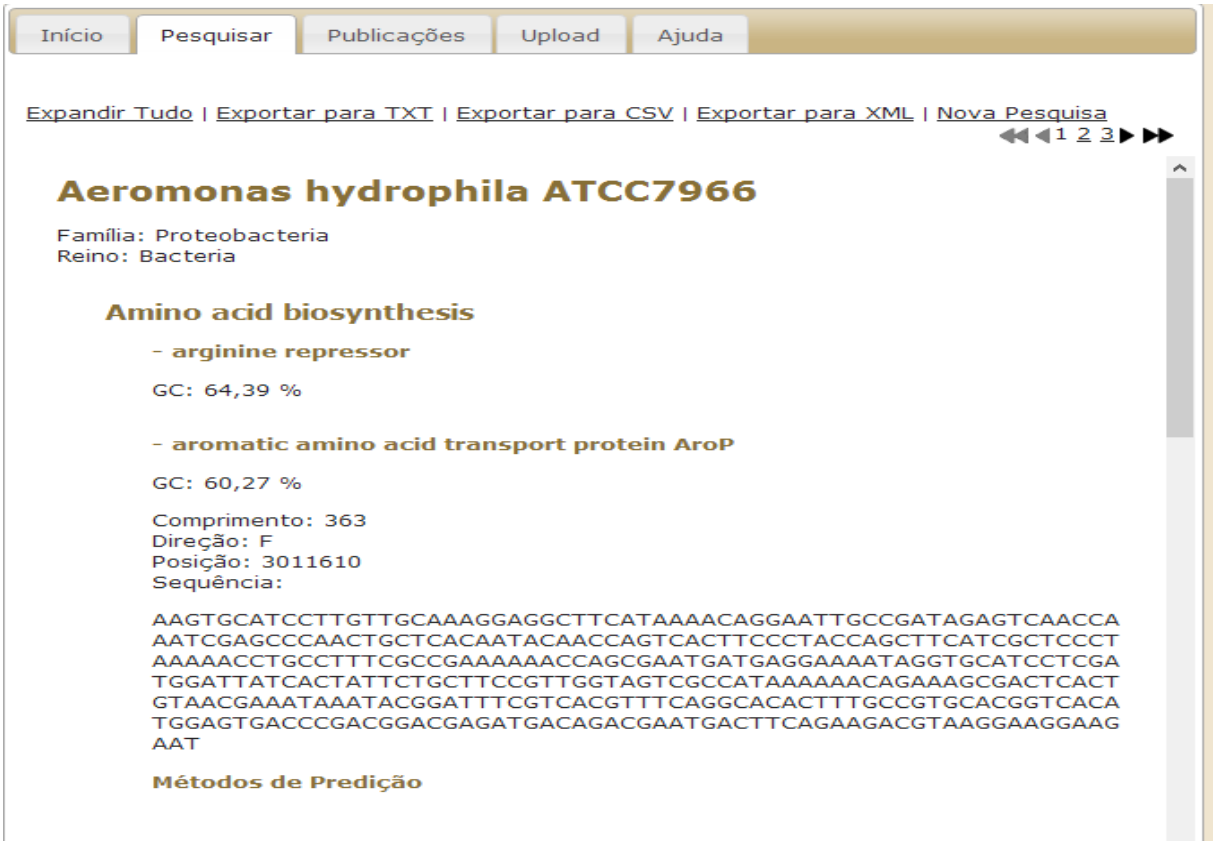

Figura 13 - Consulta no IntergenicDB

Fonte: [23]

3.3 CONSIDERAÇÕES FINAIS

O portal surgiu a partir de uma necessidade do grupo de Bioinformática da Universidade de Caxias do Sul (UCS). Seu desenvolvimento se deu através do resultado das pesquisas realizadas pelos alunos Aurione (MOLIN,2009), Vanessa (DAVANZO,2010) e Douglas (PICOLLOTO, 2012) para suas respectivas monografias de conclusão do curso. Entretanto o portal e as consultas realizadas por ele precisam passar por teste. Além da necessidade de teste que devem ser realizados no portal, surgiram novas demandas através da Profa. Dra. Scheila de Ávila e Silva que precisam ser atendidas. Os requisitos levantados a partir dessas novas demandas e os testes que devem ser realizado no IntergenicDB são o motivo deste trabalho. A modelagem e o desenvolvimento dessas novas demandas serão mostrados nas próximas seções.

[23] http://danlian.com.br/intergenicdb/Default/Search

4 PROPOSTA DE SOLUÇÃO

Este capítulo apresenta as alterações que serão feitas no banco de dados PromotoresDB e no portal IntergenicDB. Serão apresentados os novos requisitos que devem ser implementados no portal juntamente com a apresentação da nova ferramenta de importação de dados seguindo a metodologia de Desenvolvimento Guiado por Teste.

4.1 DESENVOLVIMENTO GUIADO POR TESTE

O Desenvolvimento Guiado por Teste (TDD) é um estilo de desenvolvimento de software ágil onde os desenvolvedores criam testes automatizados que são executados repetidamente durante o desenvolvimento (BECK, 2010). O uso do TDD resulta numa melhor compreensão e uma melhor gerência de códigos mais complexos (BECK, 2010). Falhas são identificadas de forma mais rápida, já que o programador tem *feedback* frequente através da execução constante dos testes (BECK, 2010).

Existem algumas etapas que devem ser seguidas em um determinado fluxo (Figura 14) no desenvolvimento orientado a teste que são (BECK, 2010):

1. Adicionar um teste rapidamente.
2. Rodar todos os teste e ver o mais novo falhando.
3. Fazer uma pequena mudança.
4. Rodar todos os teste e ver todos funcionando.
5. Refatorar e remover duplicações

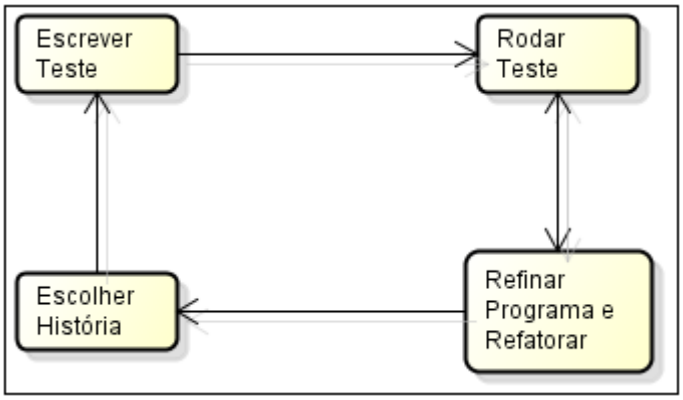

Figura 14 - Fluxo do TDD

Fonte: (BECK, 2010).

Esse ciclo é conhecido como "Vermelho-Verde-Refatora" ("Red-Green-Refactor"), onde (BECK, 2010):

- O vermelho significa que o teste está falhando, porém nesse caso a falha é tratada com um progresso, pois é possível ter uma medida concreta de falha. Todavia, é necessário arrumá-lo o mais rápido possível;
- O verde significa que o teste foi executado com sucesso;
- A refatoração, que tem como objetivo manter um código limpo e que sempre funcione.

Os casos de testes estão relatados junto com a explicação das melhorias e alterações a serem feitas no banco de dados PromotoresDB e no portal IntergenicDB, e também a explicação do importador de dados para carga de novos dados no PromotoresDB.

4.2 PROMOTORESDB

O modelo ER existente atualmente para o banco de dados PromotoresDB não está representado de forma correta a estrutura que está sendo utilizada no banco de dados. Entre as inconsistências existentes estão uma coluna denominada *IntergenicRegionLength* pertencente a tabela *ImportQueueEntry* e outra coluna denominada *Year* pertencente a tabela *Publication*. Essas colunas existem no modelo ER, porém não estão sendo utilizadas pelo PromotoresDB. É necessário analisar o portal IntergenicDB, verificar os arquivos e métodos utilizados para realizar o processo de importação de dados e verificar se as colunas *IntergenicRegionLength e Year* estão sendo empregados em algum momento neste processo de importação. Caso os campos sejam localizados então deve ser feita a criação das colunas no banco de dados, caso contrário, o modelo ER deve ser alterado, removendo os campos *IntergenicRegionLength* e *Year*. Esse processo é necessário para manter a estrutura do banco de dados PromotoresDB igual ao modelo ER.

Também será necessário alterar a cardinalidade existente hoje entre as tabelas *Gene* e *IntergenicRegion*, Atualmente, um gene pode existir sem uma

região intergênica associada a ele e uma região intergênica pode esta associada a somente um gene no máximo. Com as alterações executadas no modelo ER, um gene passa a ter obrigatoriamente uma região intergênica associada a ele e uma região intergênica poderá estar contida em mais de um gene. Com essa alteração de cardinalidade, deve-se alterar a composição da chave estrangeira da tabela gene, acrescentando a coluna *IntergenicRegionID* pertencente a tabela *IntergenicRegion*.

Outra alteração necessária é a criação de uma nova tabela denominada *Slice*. Essa nova tabela deve ser uma agregação das tabelas *IntergenicSlice*, *IntergenicRegion* e *User.* Além de colunas com as chaves primárias de cada uma das tabelas, deve ser adicionado uma nova coluna denominada *size*.

O novo modelo ER contendo as modificações citadas anteriormente podem ser visto na Figura 15.

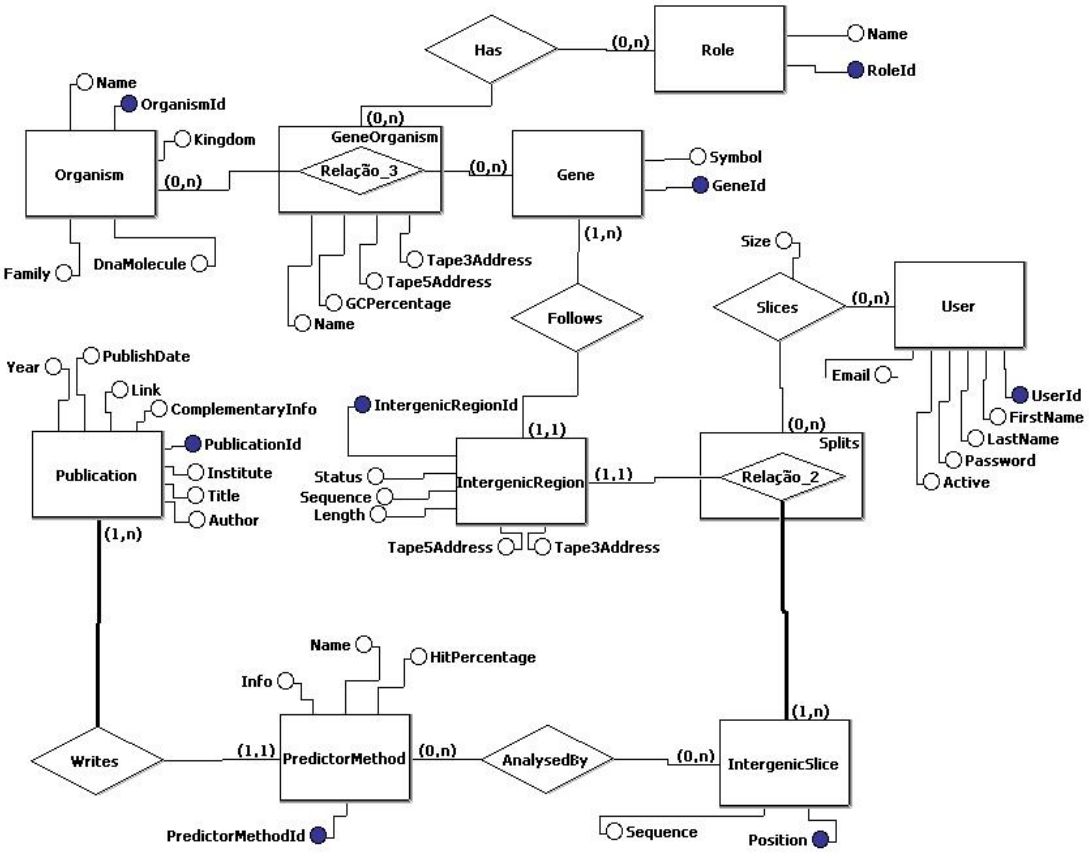

Figura 15 - Modelo Conceitual do Banco de Dados

4.3 INTERGENICDB

Nesta seção serão apresentados os requisitos levantados que devem ser implementados no portal IntergenicDB. O caso de uso com os requisitos levantados e os atores envolvidos pode ser visto na Figura 16.

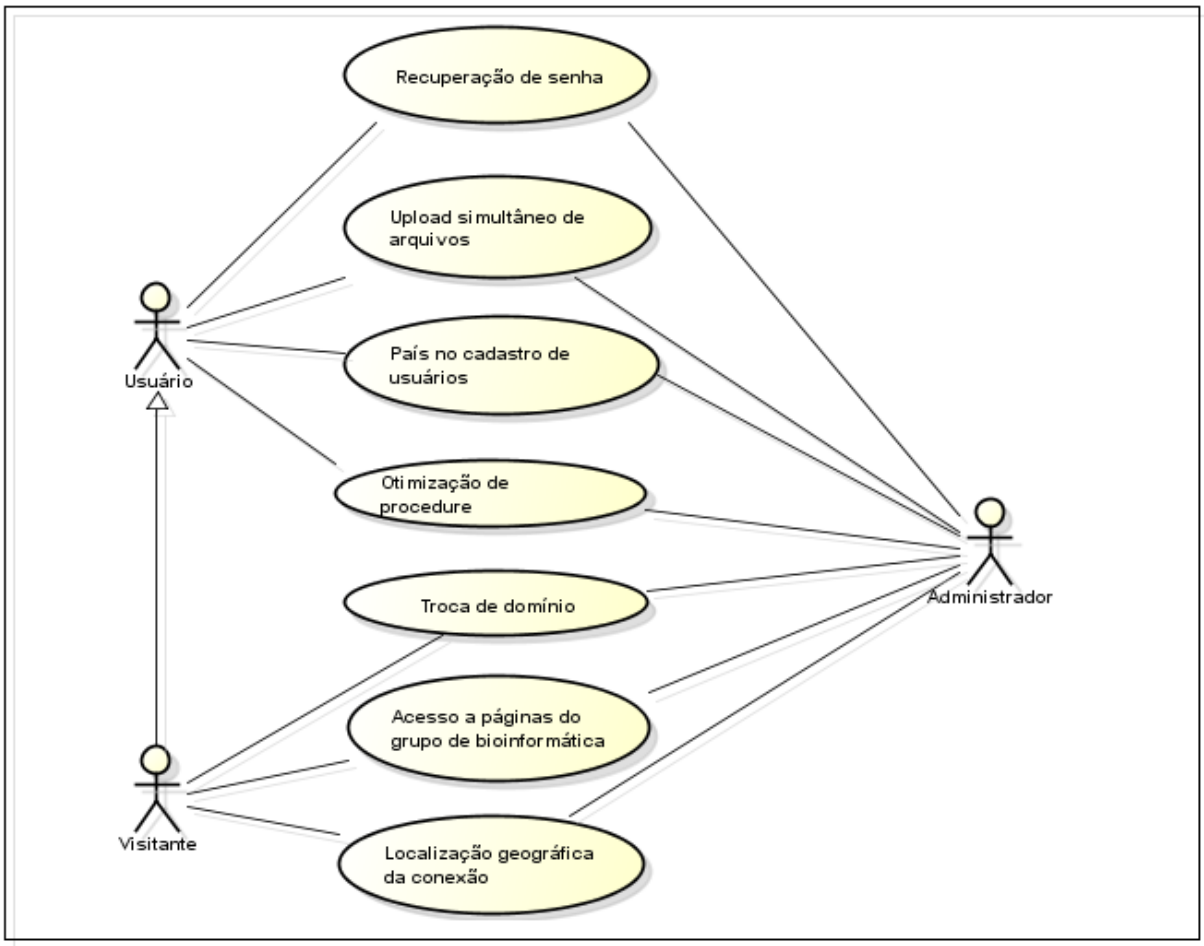

Figura 16 - Caso de uso das novas funcionalidades

4.3.1. Alterações do ER do portal

Para atender aos novos requisitos do portal IntergenicDB que serão apresentados na próxima seção, deve-se realizar a criação de uma nova tabela denominada *"Connection"* essa tabela deverá possuir as seguintes colunas:

- Coluna *"ConnectionID"* como chave primária;
- Coluna "*IPAddress"*;
- Coluna AccessDateTime;

- Coluna *“CountryOriginID”* como chave estrangeira da coluna *“CountryID”* pertencente a tabela *“Country”*.

Além da criação de uma nova tabela, deverá ser adicionada uma nova coluna na tabela *“User”* chamada *“CountryID”*.

O modelo ER atualizado com as alterações propostas pode ser visto na Figura 17.

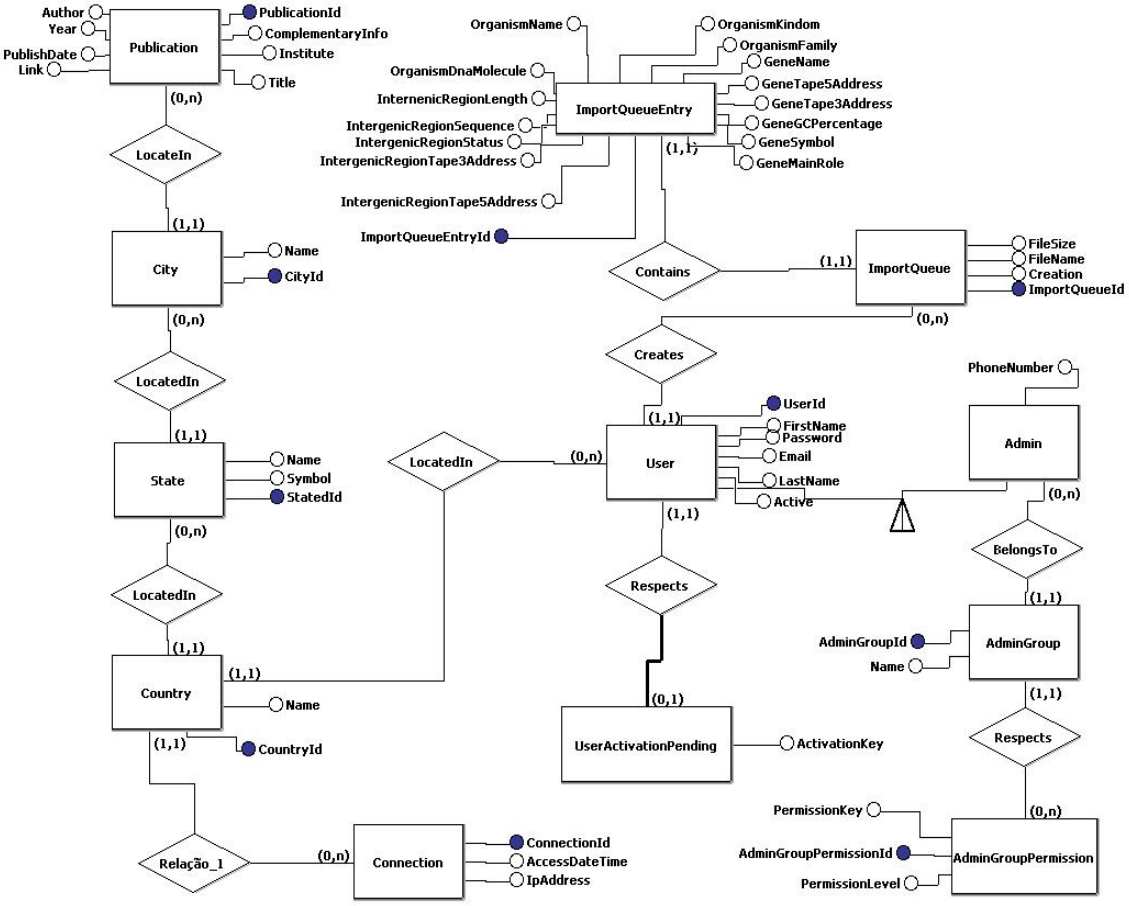

Figura 17 - Modelo ER do portal

4.3.2. Recuperação de senha

Atualmente o portal IntergenicDB permite que os usuários que acessam o portal façam um cadastro. Esse cadastro dá permissão para o usuário fazer *uploads* de arquivos para a base de dados do IntergenicDB, além de manter o acesso as informações que um usuário não cadastrado possui. Após cadastrar um usuário, é possível torná-lo um administrador, dessa forma ele adquire a permissão para acessar o painel de administração citado anteriormente. Neste painel, um

administrador, entre outra coisas, pode alterar o cadastro informações de outros usuários cadastrados, inclusive de outros administradores.

Entretanto, caso o usuário esqueça a sua senha, a única forma de recuperá-la é acessando o painel de administração e alterando a senha diretamente no cadastro do usuário. Nessa situação, é necessário que outro usuário do tipo administrador acesse o cadastro do usuário que esqueceu sua senha.

Como solução para essa situação, será criado um processo de recuperação de senha. Para ter acesso a esse novo processo, será alterada a *home* do IntergenicDB, acrescentando um novo *link* imediatamente abaixo do *link* já existente para cadastro de usuários conforme pode ser visto em destaque na Figura 18.

Figura 18 - Exemplo de nova *home*

Ao clicar no *link* criado anteriormente, uma nova tela se abrirá (Figura 15), solicitando ao usuário informar o *e-mail* utilizado no cadastro da conta que deseja recuperar a senha. Ao clicar em “Ok”, uma mensagem contendo a senha correspondente ao e-mail informado será enviada.

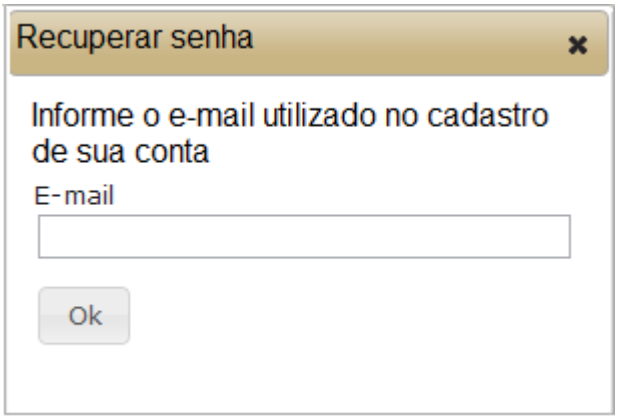

Figura 19 - Exemplo de tela para recuperação de senha

Para testar a funcionalidade, será informado um *e-mail* com a formatação inválida, obtendo assim a barra vermelha do TDD. A partir desta falha será construído o código para atingir a barra verde.

4.3.3. Armazenar e visualizar localização geográfica da conexão

Atualmente o portal IntergenicDB não possui nenhuma restrição para realizar consultas. Qualquer usuário, independente de localização geográfica pode acessar o portal. Tal liberdade de consulta aos dados permanecerá como atualmente. Porém, para fins estatísticos, é necessário armazenar no banco de dados alguns dos usuários que acessam os dados do IntergenicDB. Entre os dados que serão armazenados estão:

- Endereço IP;
- Data de acesso;
- Hora de Acesso
- País de origem.

Para possibilitar o armazenamento desses dados, será criada uma nova tabela no banco de dados PromotoresDB. A consulta desses dos acontecerá através de uma nova tela. O acesso a essa tela estará disponível através de um novo *link na* área de administração do portal IntergenicDB (Figura 20), na seção Outros.

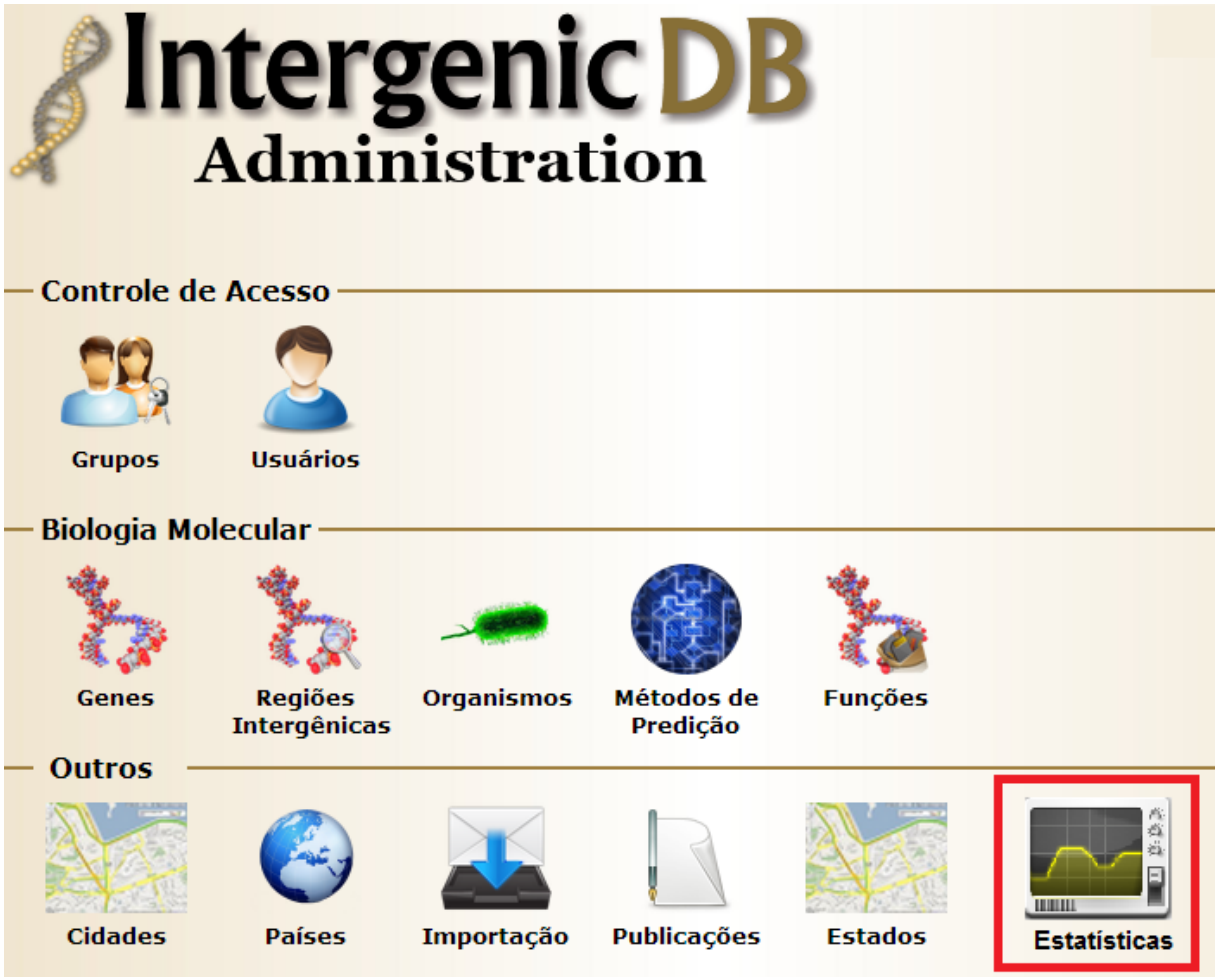

Figura 20 - Exemplo de acesso a tela de estatísticas

Ao acessar a tela de estatísticas, será exibido um relatório contendo uma contagem de acessos agrupados por país.

Para validar a funcionalidade, deve-se realizar uma consulta no banco de dados e armazenar a quantidade de registro de total de acesso ao portal. Em seguida deverá ser simulado um novo acesso ao IntergenicDB. Se a contagem de acessos efetuados teve o total acrescido em uma quantidade o teste ocorreu com sucesso.

4.3.4. Área de acesso a outros portais do grupo

O grupo de Bioinformática da Universidade de Caxias do Sul (UCS) possuirá outro portal nos moldes do IntergenicDB que ainda está em desenvolvimento. É necessário que os usuários que acessam o IntergenicDB tenham conhecimento da existência desse novo portal e de outros que futuramente possam ser criados pelo

grupo de bioinformática da UCS. Dessa forma, serão avaliadas duas opções de solução.

A primeira solução consiste na criação de uma nova aba na página inicial do IntergenicDB (Figura 21) contendo um *link* de acesso ao novo portal. Atualmente será criada apenas a área de acesso, deixando para um momento posterior à conclusão do novo portal a criação do *link* de acesso juntamente com uma breve descrição desse portal.

Figura 21 - Página do grupo

A segunda solução é adicionar a descrição deste novo portal juntamente com seu *link* de acesso na aba *"Início"*, ao final do texto já existente. Assim como na primeira solução, o *link* só estará efetivamente funcional quando o novo portal estiver concluído.

Para validar a funcionamento, será criada a nova aba com um *link*, a partir da criação desse *link* será desenvolvido o acesso ao novo portal do grupo de bioinformática da UCS, o resultado deve ser o mesmo de digitar o endereço diretamente em um navegador.

4.3.5. Upload simultâneo de arquivos

Hoje é possível carregar arquivos para o IntergenicDB. Para isso, o usuário deve fazer seu *login* no portal, acessar a área de *upload* (Figura 22) e selecionar o arquivo que deseja carregar para o IntergenicDB.

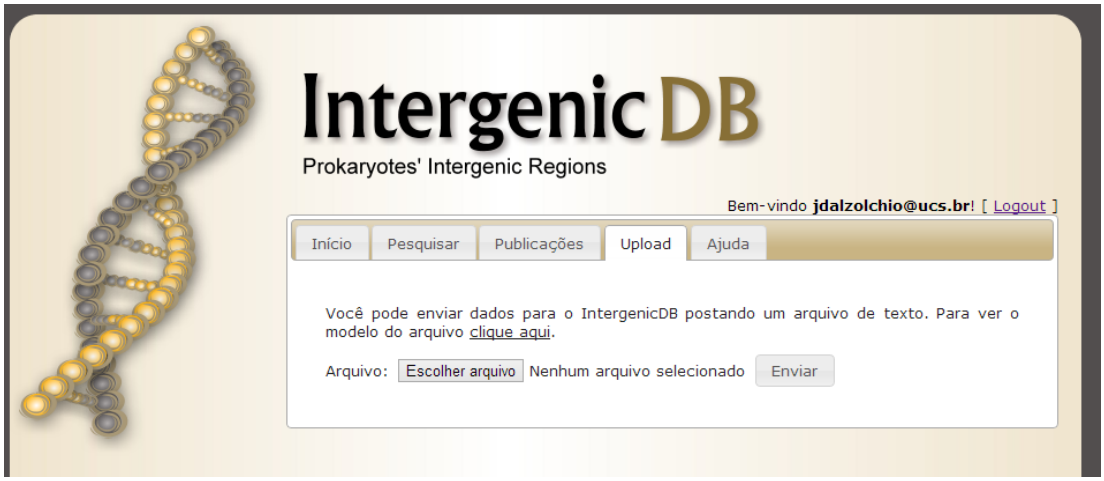

Figura 22 - *Upload* de arquivos

Com a solução atualmente disponível, é possível fazer a carga somente de um arquivo por vez. Quando existe a necessidade de importar um volume maior de arquivos, o processo acaba se tornando lento e repetitivo, além de gerar possíveis problemas, como a maior probabilidade de um arquivo passar despercebido e não ser carregado no banco de dados.

Como solução para essa situação, o *upload* de arquivos passará a contar com uma nova forma de escolha dos arquivos. Na área onde atualmente se encontra a possibilidade de enviar arquivos, será acrescentado um novo campo. Neste campo será possível escolher o diretório onde se encontram os arquivos para importação (Figura 23).

Figura 23 - *Upload* de arquivo através de diretório

Como teste para essa nova funcionalidade, será criada a importação através de um diretório inválido e de um diretório com arquivos que não estejam no formato aceito pelo portal. Ao consistir essas situações, será atingida a barra verde do TDD.

4.3.6. Cadastro de usuários

Para ter acesso a algumas funcionalidade do portal, como a área de *upload* de arquivos, é necessário que o usuário esteja previamente cadastrado no IntergenicDB. Para se cadastrar no portal é necessário que o usuário preencha alguns dados (Figura 24).

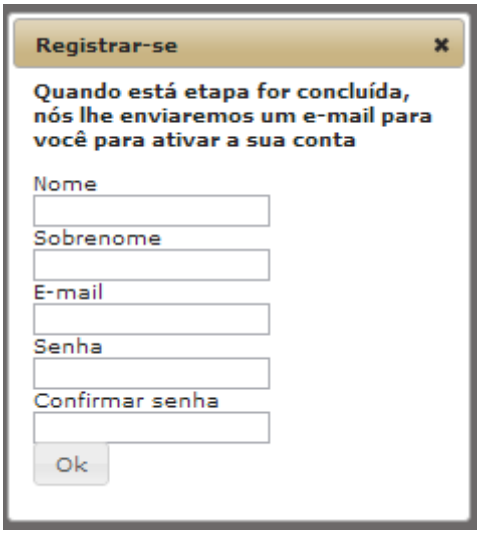

Figura 24 - Cadastro de usuários

Para fins estatísticos, será criado um novo campo na tela de cadastro de usuários (Figura 25). Esse novo campo será destinado a informar o país de origem do novo usuário. Para habilitar esse novo campo será necessário criar uma nova coluna denominada Country na tabela *User d*o banco de dados PromotoresDB para armazenar esse novo dado.

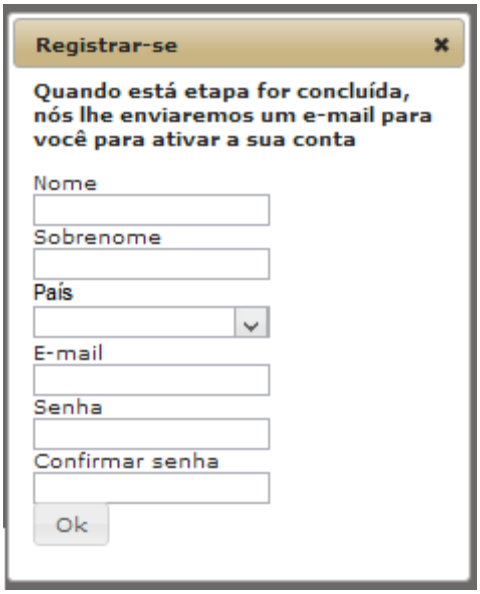

Figura 25 - Cadastro de usuários com país

Para garantir o funcionamento do novo campo, deve ser testada a navegação entre os campos em tela, além de tentar cadastrar um usuário sem informar um país válido. Ao final dos testes, a informação do país deve estar cadastrada corretamente no banco de dados.

4.3.7. Otimização de *procedure*

Atualmente existe uma *procedure* no banco de dados chamada ImportData. Essa *procedure* é responsável por inserir alguns dados em determinadas tabelas do banco de dados. Esses estão na tabela *ImportQueueEntry*, que pertence ao mesmo banco de dados, ficando a cargo do *ImportData* replicar essas informações de uma tabela para outra. Porém, conforme a quantidade de dados já armazenado no banco de dados for aumentando, o tempo de execução da *procedure* aumenta junto.

A *procedure* é executada quando é feita a confirmação de importação de algum arquivo através da tela de manutenção de importação (Figura 26). Atualmente essa execução demora mais de 60 segundos.

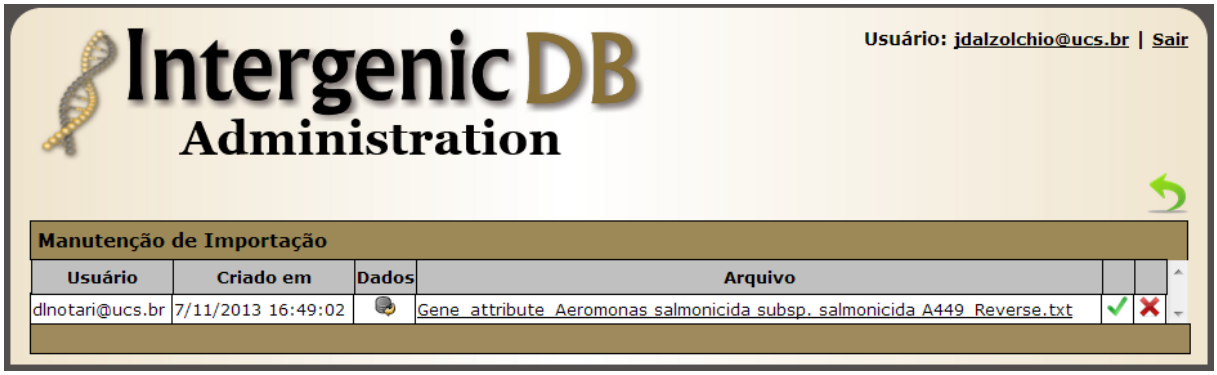

Figura 26 - Tela de manutenção de importação

Para garantir que as alterações efetuadas estejam funcionando de forma adequada, deve ser importado um arquivo utilizando *procedure* já existente e posteriormente executar a mesma importação utilizando a *procedure* que esteja otimizada. Ao final do processo, os dados importados devem ser os mesmos e a nova *procedure* deve ter finalizada a importação no menor tempo possível para garantir uma usabilidade mais adequada ao usuário.

4.3.8. Troca de domínio

Atualmente o portal IntergenicDB está hospedado em um domínio particular, pertencente a um dos membros do grupo de bioinformática da UCS. Como o número de pessoas envolvidas com a manutenção do portal vem aumentando, utilizar um domínio que possua informações particulares que não devem ser compartilhadas pode ser tornar um problema.

A solução proposta é a criação de um novo domínio que não esteja vinculado aos dados particulares de nenhum membro do grupo de bioinformática da UCS. Para que o novo domínio esteja funcional, é necessário migrar o banco de dados do portal e o próprio portal IntergenicDB deve ser realocado para o novo domínio.

Como forma de garantir que o funcionamento do IntergenicDB permaneça inalterado, algumas consultas serão feitas no domínio atual e repetidas no novo domínio. O resultado final deve ser o mesmo.

Neste processo, todas as classes do portal IntergenicDB serão afetadas, já que a troca de domínio implicará na migração dos arquivos do portal para um novo diretório de acesso.

4.4 IMPORTADOR DE DADOS

Hoje o portal disponibiliza uma área onde é possível carregar arquivos com dados de biologia molecular (Figura 27), desta forma é possível importar uma grande quantidade de informações para o banco de dados de forma automática. Porém, a geração desses arquivos é feita de forma manual em uma interface caractere, utilizando scripts desenvolvidos na linguagem de programação *python*, que acaba gerando apenas um arquivo de um organismo por vez.

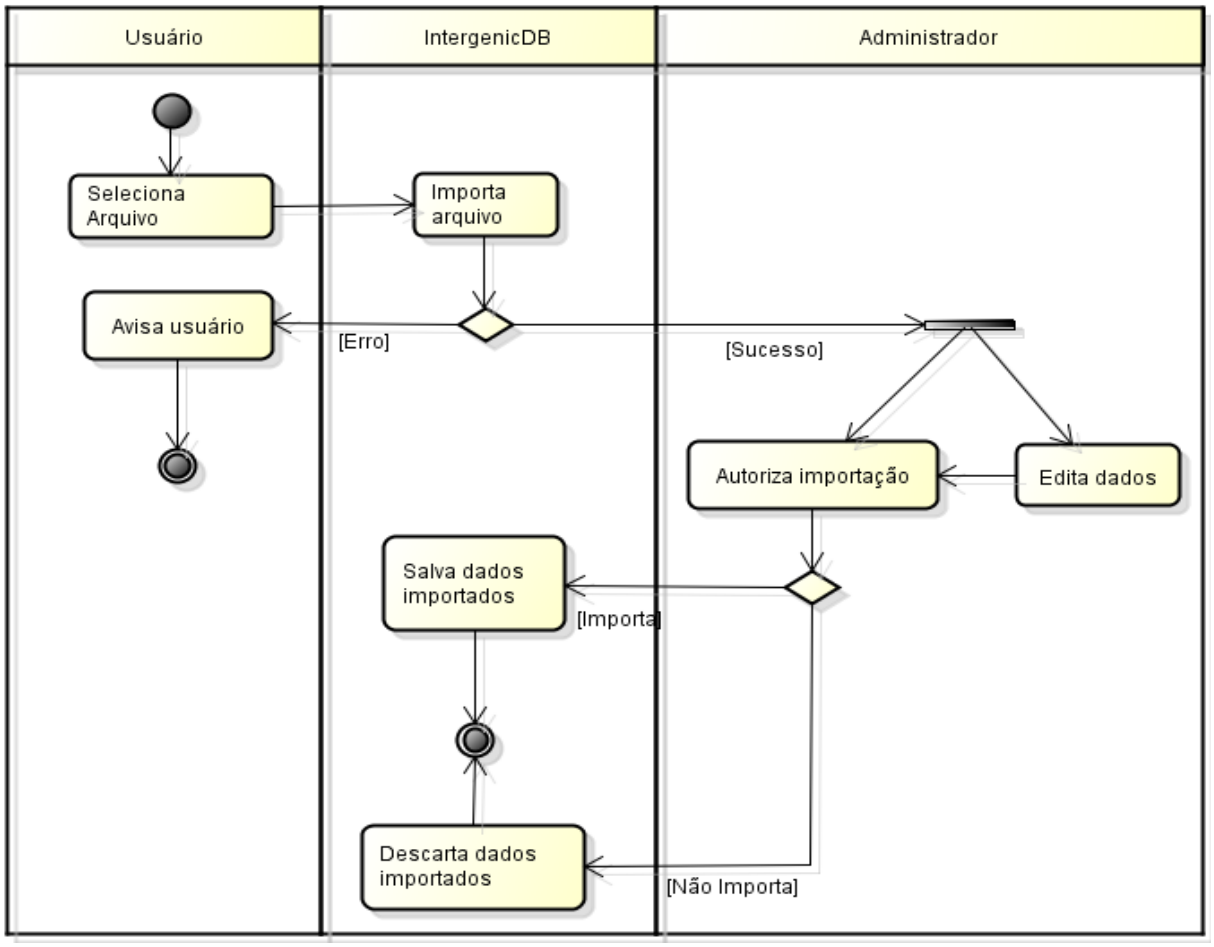

Figura 27 - Processo de importação de dados do IntergenicDB

Como solução para automatizar o processo de geração de arquivos para serem importados pelo IntergenicDB será criada uma ferramenta utilizando uma interface gráfica. Essa nova interface deve seguir o modelo de tela como mostrado na Figura 28.

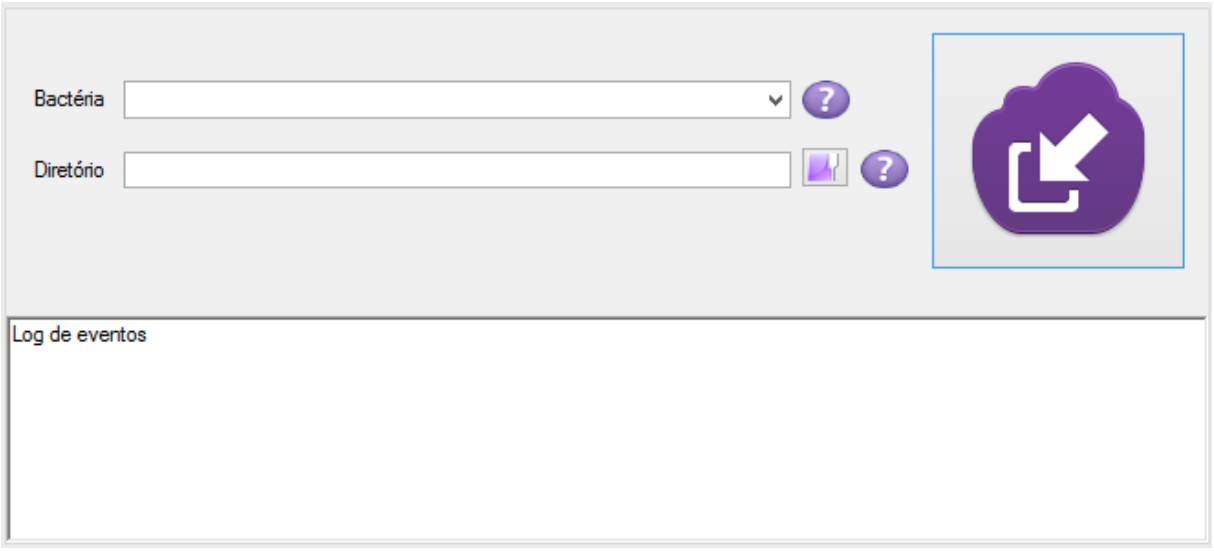

Figura 28 - Protótipo de tela para o importador de dados

Essa ferramenta deve ser capaz de se conectar com o banco de dados do NCBI e do JCVI – CMR e fazer o *download* dos dados contidos em cada um dos bancos de dados. Os dados obtidos devem ser cruzados gerando um novo arquivo para cada organismo. As informações serão cruzadas da seguinte forma:

- O arquivo deve ser gerado utilizando o leiaute aceito pelo IntergenicDB para realizar a importação das informações.
- Serão criados dois arquivos para cada organismo, um com informações no sentido *"forward" e* outro com as informações no sentido *"reverse"*.
- Será dada prioridade aos dados pertencentes ao NCBI. Somente será feita a procura de uma sequência intergênica no JCVI – CMR se esta existir no NCBI.
- O usuário dessa nova ferramenta poderá escolher se deseja consultar todos os organismos em cada um dos bancos ou se deseja fazer o *download* dos dados de um organismo específico. Os arquivos gerados pelo importador de dados serão salvos em um diretório previamente selecionado pelo usuário.

Após realizar o processo de geração de arquivos através do importador (Figura 29) o usuário utilizará portal IntergenicDB para realizar o *upload* dos dados para o banco de dados. Para auxiliar no processo de *upload*, será utilizada a nova funcionalidade de *upload* simultâneo de arquivos citada na seção anterior.

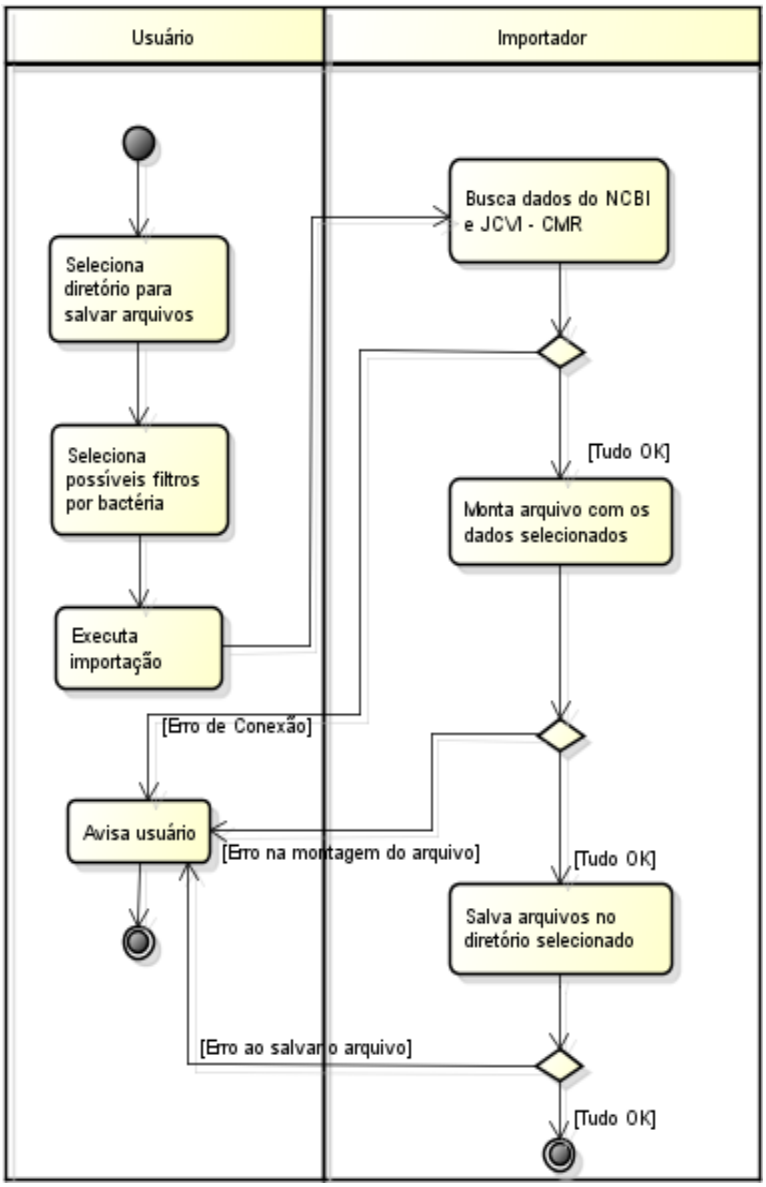

Figura 29 - Processo do importador de dados

Essa funcionalidade será desenvolvida para possibilitar a importação dos dados informando apenas o diretório onde se encontram os arquivos. Para validar o funcionamento adequado desta ferramenta, deve-se criar um arquivo manualmente ou utilizar um arquivo já gerado utilizando os scripts em *python* atualmente empregados para a geração dos arquivos. Com os arquivos criados será feita uma comparação com os arquivos gerados pelo importador de dados. Os arquivos gerados pelo importador devem estar iguais aos arquivos criados incialmente para realizar a comparação.

4.5 CONSIDERAÇÕES FINAIS

Com a criação de novas ferramentas de auxílio, o IntergenicDB possuirá um ambiente mais ágil para navegação e população do banco de dados. Deverão ser implementadas algumas alterações no portal e criação de uma nova ferramenta que serão de fundamental importância para atingir os resultados esperados.

No próximo capítulo será descrito o desenvolvimento dos requisitos apresentados neste capítulo, apresentado características da implementação, as dificuldades encontradas entre outras ações tomadas no processo de desenvolvimento.

5 DESENVOLVIMENTO

Neste capítulo serão apresentadas as principais etapas da implementação dos requisitos levantados no capítulo 4. Descrevendo também as dificuldades encontradas e as atitudes tomadas para contorná-las.

5.1 REQUISITOS DO INTERGENICDB

Nesta seção será apresentado o desenvolvimento dos requisitos do portal IntergenicDB descritos no capítulo 4 (seção 4.3).

5.1.1 Alterações do ER do portal

As novas colunas necessárias para armazenar as informações, referentes as alterações realizadas a partir dos requisitos levantados, já estão contempladas na modelo E.R do banco de dados do portal IntergenicDB. A criação das novas colunas foi feita através de comandos excetuados diretamente no SGBD.

5.1.2 Recuperação de senha

A interface desenvolvida para a recuperação de senha se manteve conforme descrito no requisito levantado para esse projeto (seção 4.3.2). Porém, devido à forma como o portal IntergenicDB armazena a senha dos usuários no banco de dados, não é possível recuperá-la e fazer o seu envio por e-mail.

Como alternativa a essa característica do IntergenicDB, foi desenvolvida uma nova solução. Ao solicitar a recuperação de senha, o usuário irá receber em seu e-mail um código que servirá como nova senha. Para a recuperação de senha foi criada um nova operação no *controller*[24] *AccountController* (Figura 30) chamada de *ForgotPassoword*.

[24] Controller: Elemento conceitual do padrão MVC (Model –View-Controller)

```
[AcceptVerbs(HttpVerbs.Post)]
public ActionResult ForgotPassword(ForgotPasswordViewModel model)
{
    if (string.IsNullOrEmpty(model.Email))
        ModelState.AddModelError("Email", GlobalizationHelper.GetLocableString("SignUp_EmptyEmail"));

    if (ModelState.IsValid)
    {
        try
        {
            using (var gateway = DataGatewayFactory.GetDataGateway<User>())
            {
                User user = new User();
            A   user = gateway.Find(Predicate.Where("Email").IsEqualTo(model.Email)).First();
                if (user != null)
                {
                    EmailService email = new EmailService();
                    String newpassword = CryptographyHelper.EncodePassword(DateTime.Now.ToString());
                    user.Password = CryptographyHelper.EncodePassword(newpassword);
                B   gateway.Save(user);
                    email.SendEmail(model.Email, "IntergenicDB",  GlobalizationHelper.
                                    GetLocableString("ForgotPassword_Email") + ": " + newpassword);
                }
            }
            return RedirectToAction("Index", "Default");
        C }
        catch
        {
            return RedirectToAction("Index", "Default");
        }
    }
    else
    {
    D   return View("Default", model);
    }
}
```

Figura 30 - Detalhes do código desenvolvido para recuperação de senha

Esta nova operação recebe a informação de *e-mail* para recuperação da senha através do argumento *"model"*. Com base no *e-mail* informado, é verificado se existe algum usuário cadastrado com esse *e-mail* (A). Caso o usuário exista, então é criada uma nova senha que substitui a atual e é enviada para o e-mail informado (B). Ao final do processo, o usuário é redirecionado para a página inicial do IntergenicDB. Para garantir o correto funcionamento da geração senhas foi criado o método de teste *CreateNewPassword* (Figura 31).

```
[Test]
public void CreateNewPassword()
{
    using (var gateway = DataGatewayFactory.GetDataGateway<User>())
    {
    (A) User user = gateway.Find(Predicate.Where("UserID").
                    IsNotEqualTo(0)).First();
        if (user==null)
        {
            user = new User()
            {
                Active = true,
                Country = null,
                CountryId = 1,
                Email = "bioinfoucs@gmail.com",
                FirstName = "FirstName",
                LastName = "LastName",
                Password = "c4ca4238a0b923820dcc509a6f75849b"

            };
        (B) gateway.Save(user);
        }
    (C) String newpassword = EncodePassword(DateTime.Now.ToString());
        user.Password = EncodePassword(newpassword);
        gateway.Save(user);
    (D) Assert.AreEqual(gateway.Find(Predicate.Where("Password").
                                IsEqualTo(user.Password)).Count(), 1);
    }

}
```

Figura 31 - Detalhe do código para geração de novas senhas

O teste busca um usuário qualquer do banco (A), caso não exista usuários no banco de dados, é criado um novo (B). É gerada uma nova senha criptografada para esse usuário (C). Ao final do teste, é feita uma pesquisa no banco de dados utilizando a nova senha como filtro (D), caso o banco retorne um registro, significa que o teste foi executado com sucesso.

5.1.3 Armazenar e visualizar localização geográfica da conexão

Para a tela responsável por visualizar as informações de cada novo acesso realizado ao portal foi utilizada a estrutura já aplicada para outras telas da área de manutenção (PICOLLOTO, 2012), onde cada tela possui uma *key*, e com base nessa *key*, é criado um *Gateway (DAVANZO, 2010)*. Ao final do processo, é efetuada uma consulta paginada, retornando 14 registros para serem exibidos na primeira página.

Para armazenar as informações de acesso foi realizada uma alteração na operação *Index* (Figura 32) do c*ontroller DefaultController*, que por padrão, é a primeira *operação* a ser executada ao acessar o portal IntergenicDB.

```
public ActionResult Index()
{
    Connection connection = new Connection();
    IPAddress[] localIPs = Dns.GetHostAddresses(Dns.GetHostName());
    foreach (IPAddress addr in localIPs)
    {
        if (addr.AddressFamily == AddressFamily.InterNetwork)
        {
            connection.IPAddress = addr.ToString();
        }
    }
    string coutry = CultureInfo.CurrentCulture.EnglishName.
                    Substring(CultureInfo.CurrentCulture.EnglishName.IndexOf('(') + 1,
                    CultureInfo.CurrentCulture.EnglishName.LastIndexOf(')') -
                    CultureInfo.CurrentCulture.EnglishName.IndexOf('(') - 1);
    using (var gateway = DataGatewayFactory.GetDataGateway<Country>())
    {
       Country c = gateway.Find(Predicate.Where("Name").IsEqualTo(coutry)).FirstOrDefault();
       if (c == null)
       {
           c = new Country();
           c.Name = coutry;
           gateway.Save(c);

       }
       connection.CountryId = c.CountryId;
    }
    connection.AccessDateTime = DateTime.Now;
    using (var gateway = DataGatewayFactory.GetDataGateway<Connection>())
    {
        gateway.Save(connection);
    }
    return View();
}
```

A
B
C
D
E

Figura 32 - Detalhes do código para armazenar informações da conexão

Quando um usuário acessa o portal, seu endereço de IP (A) e seu país (B) são armazenado juntamente com a data e hora do acesso. Caso o país de origem da conexão não exista, o portal cria um novo país (C) e então recupera seu ID. Ao final da operação *Index* as informações de conexão são salvas no banco de dados (E).

O teste deste requisito (Figura 33) foi realizado no método *InsertNewConnection* (A). Uma nova conexão é criada (B) e o número atual de acessos no banco de dados é armazenado em uma variável local (C). A conexão é salva no banco de dados e imediatamente recuperada (D). O teste valida se a conexão foi salva com sucesso (E) para então ser apagada do banco de dados (F).

```
[Test]
Ⓐ public void InsertNewConnection()
{
    using (IDataGateway<Country> gatewaycountry = DataGatewayFactory.
                                                  GetDataGateway<Country>())
    {
        Country country = gatewaycountry.Find(Predicate.Where("CountryId").
                                              IsNotEqualTo(0)).FirstOrDefault();
        GetCountryId
        using (var gateway = DataGatewayFactory.GetDataGateway<Connection>())
        {
            Connection connection = new Connection()
            {
              Ⓑ AccessDateTime = DateTime.Now, CountryId= country.CountryId,
                Country = country, IPAddress = "999.999.999.999"

            };
            Assert.That(connection.ConnectionID, Is.EqualTo(0));
          Ⓒ long connectionnumber = gateway.Find(Predicate.Where("ConnectionID").
                                        IsNotEqualTo(0)).
                                        Count();
            gateway.Save(connection);
            Assert.That(connection.ConnectionID, Is.GreaterThan(0));
          Ⓓ Connection connectionsaved = gateway.Find(Predicate.Where("ConnectionID").
                                                IsEqualTo(connection.ConnectionID)).
                                                FirstOrDefault();
          Ⓔ Assert.AreEqual(connection.ConnectionID  , connectionsaved.ConnectionID);
            Assert.AreEqual(connectionnumber + 1, gateway.Find(Predicate.Where("ConnectionID").
                                                  IsNotEqualTo(0)).Count());

          Ⓕ gateway.Delete(connection);
        }
    }
}
```

Figura 33 - Detalhe do código de teste de nova conexão

5.1.4 Área de acesso a outros portais do grupo

Para este requisito foi adota a segunda solução proposta. Neste caso, as informações e acessos a possíveis outros portais do grupo de Bioinformática da UCS serão adicionados ao final do texto da tela de boas-vindas do portal IntergenicDB (Figura).

Bem-vindo ao IntergenicDB!

O IntergenicDB é um banco de dados que contém sequências de DNA conhecidas como regiões intergênicas (ou promotores) de seres procariotos.

Este banco de dados é um resultado do estudo efetuado pela Prof. Sheila de Ávila e Silva, da Universidade de Caxias do Sul (UCS), entitulado "Redes Neurais Artificiais no Reconhecimento de Regiões Promotoras em Bactérias Gram-Negativas". Este estudo investiga as regiões intergênicas, utilizando redes neurais treinadas por um programa para identificar se dada sequência de DNA é uma região promotora. Para ver mais sobre este estudo, clique aqui.

O website IntergenicDB é uma ferramenta que prove uma forma simples de efetuar pesquisar e enviar dados para o banco de dados do IntergenicDB. Este website é um resultado do estudo dos alunos da Universidade de Caxias do Sul Vanessa Davanzo e Douglas Picolotto, com a orientação do Prof. Daniel Luis Notari.

Clique aqui para saber mais sobre o grupo de pesquisa de Bioinformática da Universidade de Caxias do Sul.

Figura 34 - Tela boas-vindas do IntergenicDB

5.1.5 Upload simultâneo de arquivos

Diferente do que foi proposto inicialmente, não será possível informar um diretório para *upload* de arquivos. Para atender a necessidade de poder selecionar mais de um arquivo para a importação por vez, foi aproveitada a estrutura já existente, sem que fosse necessário modificar o leiaute da tela. A importação de múltiplos arquivos por vez ocorre quando, na tela de escolha de arquivo, mais de um arquivo é selecionado. É possível ver na Figura 35 - Detalhe do código para upload de múltiplos arquivos que foram feitas apenas duas alterações no método de *Upload*.

```
[AcceptVerbs(HttpVerbs.Post)]
public ActionResult Upload(IEnumerable<HttpPostedFileBase> file)
{
    StringBuilder sLog = new StringBuilder();
    if (!Request.IsAuthenticated)
    {
        return RedirectToAction("Index");
    }
    foreach (var f in file)
    {
        try
        {
            var user = this.SessionManager.GetSessionUser();
            var queueItem = QueueItemBuilder.BuildFrom(f, user);
            var importDirectoryPath = ConfigurationManager.AppSettings["ImportDirectoryPath"];

            ResolvePath(ref importDirectoryPath);

            using (var service = new IntergenicDatabaseService())
                service.ImportData(queueItem, importDirectoryPath);
        }
        catch
        {
            if (sLog.Length == 0)
                sLog.AppendLine(f.FileName);
            else
                sLog.AppendLine(";" + f.FileName);
        }
    }

    if (sLog.Length == 0)
    {
        return RedirectToAction("Upload");
    }
    else
    {
        ViewData["Errors"] = new List<string> {
            GlobalizationHelper.GetLocableString("Message_ErrorUpload") + "\n" + sLog.ToString()};
        return View();
    }

}
```

A
B

Figura 35 - Detalhe do código para upload de múltiplos arquivos

A primeira alteração foi modificar o parâmetro *file*, para que aceitasse receber mais de um arquivo por *upload* (A). Desta forma foi necessário apenas criar um laço de repetição (B). Assim o *upload* irá acontecer para cada arquivo da mesma forma como já ocorria anteriormente.

5.1.6 Cadastro de usuários

O desenvolvimento deste requisito se deu conforme descrito no capítulo anterior. Para garantir seu funcionamento foi criada uma classe no projeto *IntergenicDb.Data.Test* (Figura 36) denominada *UserGatewayTest* (A). Nesta nova classe foi implementado o método de teste *InsertUserWithCoutry* (B).

```
A  class UserGatewayTest
   {
       [Test]
B      public void InsertUserWithCountry()
       {
           using (IDataGateway<Country> gatewaycountry = DataGatewayFactory.
                                                          GetDataGateway<Country>())
           {
C              Country country = gatewaycountry.Find(Predicate.Where("CountryId").
                                                    IsNotEqualTo(0)).FirstOrDefault();

               if (!gatewaycountry.Exists(country))
               {
D                  Country c = new Country (){Name = "Brasil"};
                   gatewaycountry.Save(country);
               }
               using (var gateway = DataGatewayFactory.GetDataGateway<User>())
               {
E                  User user = new User()
                   {
                       Active = true,
                       Country = null,
                       CountryId = country.CountryId,
                       Email = "teste@tesc.com",
                       FirstName = "FirstName",
                       LastName = "LastName",
                       Password = "c4ca4238a0b923820dcc509a6f75849b"

                   };
                   Assert.That(user.UserId, Is.EqualTo(0));
F                  gateway.Save(user);
                   Assert.That(user.UserId , Is.GreaterThan(0));
G                  User usersaved = gateway.Find(Predicate.Where("UserId").IsEqualTo(user.UserId)).
                                                                                  FirstOrDefault();
H                  Assert.AreEqual(user.CountryId, usersaved.CountryId);
I                  gateway.Delete(user);
               }
           }
       }
   }
```

Figura 36 - Detalhe da classe de teste para gravação de usuários

O método seleciona um país qualquer no banco de dados (C). Se necessário é criado um país novo para o teste (D). Esse pais é utilizado na propriedade *CountryID* do novo usuário (E). O método tenta gravar o usuário criado (F) e busca-lo no banco (G) para então verificar se a gravação ocorreu com êxito (H). Ao final do teste, o usuário criado é removido do banco de dados (I).

5.1.7 Otimização de *procedure*

Conforme pode ser visto na Figura 37, a *procedure ImportData* transfere as informações existentes na tabela *ImportQueueEntry* para as tabelas *Organism* (A), *Role* (B), *Gene* (C) e *IntergenicRegion* (D).

```
Ⓐ insert into Organism (Name, Kingdom, Family, DnaMolecule)
     select distinct OrganismName, OrganismKingdom, OrganismFamily,
       OrganismDnaMolecule
     from
       ImportQueueEntry q
     where
       (OrganismName not in (select Name from Organism)) and q.ImportQueueId = _importQueueId;

Ⓑ insert into Role (Name)
     select distinct GeneMainRole
     from ImportQueueEntry q
     where
       GeneMainRole not in (select Name from Role) and q.ImportQueueId = _importQueueId;

Ⓒ insert into Gene (OrganismId, RoleId, Name, Symbol, Tape5Address, Tape3Address, GCPercentage)
   select distinct
      OrganismId, RoleId, GeneName as Name, GeneSymbol as Symbol, GeneTape5Address as Tape5Address,
      GeneTape3Address as Tape3Address, GeneGCPercentage as GCPercentage
   from
     ImportQueueEntry q
   inner join organism o on o.Name = q.OrganismName and o.Family = q.OrganismFamily and
             o.Kingdom = q.OrganismKingdom and o.DnaMolecule = q.OrganismDnaMolecule
   inner join role r on (r.Name = q.GeneMainRole or q.GeneMainRole is null)
  where q.ImportQueueId = _importQueueId;

Ⓓ insert into IntergenicRegion (Tape5Address, Tape3Address, Sequence, Length, Status, GeneId)
     select IntergenicRegionTape5Address as Tape5Address, IntergenicRegionTape3Address as Tape3Address,
       IntergenicRegionSequence as Sequence, q.IntergenicRegionLength as Length,
       IntergenicRegionStatus as Status, GeneId
     from
       ImportQueueEntry q
     inner join organism o on o.Name = q.OrganismName and o.Family = q.OrganismFamily and
               o.Kingdom = q.OrganismKingdom and o.DnaMolecule = q.OrganismDnaMolecule
     inner join role r on r.Name = q.GeneMainRole
     inner join gene g on g.OrganismId = o.OrganismId and g.Name = q.GeneName and
               g.RoleId = r.RoleId and g.GCPercentage = q.GeneGCPercentage and
               g.Tape5Address = q.GeneTape5Address and g.Tape3Address = q.GeneTape3Address
  where q.ImportQueueId = _importQueueId;
```

Figura 37 - *Procedure* para importação de dados

Uma das características para inserir os dados nas tabelas de destino é a existência de comandos de junção utilizando colunas alfanuméricas, sendo a

inserção na tabela *intergenicRegion* (D) a mais crítica. Além do uso de "*not in"* para realizar comparações em filtros. Essas características impactam negativamente no tempo de execução da procedure, sendo que esse tempo tende a ficar maior conforme a quantidade de dados existente no banco de dados for aumentando.

Atualmente o tempo de execução da *procedure* é, em média, de um minuto. Esse tempo torna o processo de importação no IntergenicDB significativamente lento. Pensando em otimizar a execução da *ImportData*, foram feitas algumas modificações na forma como são feitas as consultas e inserções de dados através da procedure.

No processo de análise e otimização da *procedure*, foi verificado que não existia uma validação da existência do gene e da região intergênica antes da inserção de um novo registro no banco de dados. Um mesmo arquivo importado mais de uma vez resultaria em duplicidade de informações.

Para resolver o problema de duplicidade (Figura 38), foi criado um cursor que avalia cada registro importado, verificando se ele deve ou não ser gravado no banco de dados.

```
DECLARE pOrganismID INT(10); DECLARE pRoleID INT(10);
DECLARE pGeneID INT(10); DECLARE pIntergenicRegionID INT(10);
DECLARE pIntergenicRegion INT(10); DECLARE v_finished INTEGER DEFAULT 0;
DECLARE pOrganismName varchar(256); DECLARE pRoleName varchar(512);
DECLARE pGeneSymbol varchar(512); DECLARE pGeneTape5Address INT(10);
DECLARE pGeneTape3Address INT(10); DECLARE pIntergenicRegionTape5Address INT(11);
DECLARE pIntergenicRegionTape3Address INT(11); DECLARE import_cursor CURSOR FOR
    SELECT ImportQueueEntryId, OrganismName, GeneMainRole, GeneTape5Address, GeneTape3Address, IntergenicRegionTape5Address,
        IntergenicRegionTape3Address, GeneSymbol FROM importqueueentry WHERE ImportQueueId = _importQueueId;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET v_finished = 1;
OPEN import_cursor;
get_import: LOOP
    FETCH import_cursor
    INTO pIntergenicRegion, pOrganismName, pRoleName, pGeneTape5Address, pGeneTape3Address, pIntergenicRegionTape5Address,
        pIntergenicRegionTape3Address, pGeneSymbol;
    IF v_finished = 1 THEN
    SET pOrganismID = (SELECT MAX(Organism.OrganismId) FROM Organism WHERE Organism.Name = pOrganismName);
    IF (pOrganismID IS NULL)  THEN

    SET pRoleID = (SELECT MAX(Role.RoleId) FROM Role WHERE Role.Name = pRoleName);
    IF (pRoleID IS NULL)  THEN

    SET pGeneID = (SELECT MAX(Gene.GeneId) FROM Gene WHERE Gene.Symbol = pGeneSymbol);
    IF (pGeneID IS NULL)  THEN

    if ((SELECT MAX(GeneOrganism.GeneId) FROM GeneOrganism
        WHERE GeneOrganism.GeneID = pGeneID AND GeneOrganism.OrganismID = pOrganismID) IS NULL) THEN

    if ((SELECT MAX(GeneOrganismRole.GeneId) FROM GeneOrganismRole
        WHERE GeneOrganismRole.GeneID = pGeneID AND GeneOrganismRole.OrganismID = pOrganismID
        AND GeneOrganismRole.RoleID = pRoleID) IS NULL) THEN

    SET pIntergenicRegionID = (SELECT MAX(IntergenicRegion.IntergenicRegionId) FROM IntergenicRegion
                               WHERE Tape5Address = pIntergenicRegionTape5Address
                               AND Tape3Address = pIntergenicRegionTape3Address);
    IF (pIntergenicRegionID IS NULL) THEN
END LOOP get_import;
CLOSE import_cursor;
```

Figura 38 - Nova *procedure* para importação de dados

Essa validação é feita através de consultas no banco para avaliar a existência do registro. Caso a consulta não retorne nenhuma informação do banco de dados, então é feita a inserção. Senão o resultado da consulta é armazenado em uma variável, para que em consultas posteriores seja utilizado o identificador da tabela como filtro.

Com o objetivo de estruturar melhor o banco de dados e facilitar no controle das informações que são importadas pelo IntergenicDB, foram criadas duas novas tabelas denominadas *GeneOrganism* e *GeneOrganismRole*. Além disso, a tabela Gene precisou ser reestruturada.

As colunas que armazenam a região intergênica nas tabelas *IntergenicRegion* e *IntergenicRegioQueue* precisaram ter seu tipo de dado alterado de *Varchar* para *MediumText*, já que em alguns arquivos as informações excediam o tamanho máximo permitido.

O resultado das mudanças no banco de dados pode ser visto na Figura 17 - Modelo ER do portal. Como consequência da mudança da estrutura de tabelas do banco, a *View SimpleSearch* responsável por agrupar as informações de várias tabelas, precisou ser adaptada para que a funcionalidade de pesquisa do IntergenicDB continuasse funcionando adequadamente.

5.1.8 Criação e alteração das manutenções dos dados do gene

A alteração da tabela Gene e a criação de duas novas tabelas, como citado na seção anterior, resultaram em alterações na área de administração do portal. Como foi necessário criar as telas de manutenção para as tabela *IntergenicOrganism* e *IntergenicOrganismRole*, a tela de administração do portal IntergenicDB agora conta com dois novos acessos, cada um dará acesso a um das novas tabelas. A alteração na tabela de Gene não acarretou nenhuma mudança na área administrativa.

5.1.9 Troca de domínio e versionamento do código fonte

Com a troca de domínio, o endereço eletrônico para acesso ao portal IntergenicDB foi alterado (http://intergenicdb.bioinfoucs.com). Tanto o antigo quanto

o novo domínio possuem a capacidade de servirem como servidor para versionamento de código fonte, pois possuem um sistema de controle de versão (SCV) denominado *SubVersion*. *SubVersion* é uma ferramenta *free/open source* que gerencia arquivos, diretórios e as mudanças que eles sofrem com o passar do tempo (COLLINS-SUAAMAN *et al.*, 2011). Na máquina cliente foi utilizado o *software TortoiseSVN*[25], que dispõe de uma interface gráfica para poder realizar o *download* e *upload* de arquivos e assim realizar o versionamento do código fonte. Na Figura 39 pode-se ver o endereço eletrônico utilizado para guardar os arquivos do código fonte do IntergenicDB através da ferramenta *Repository Browser* do cliente *TortoiseSVN.*

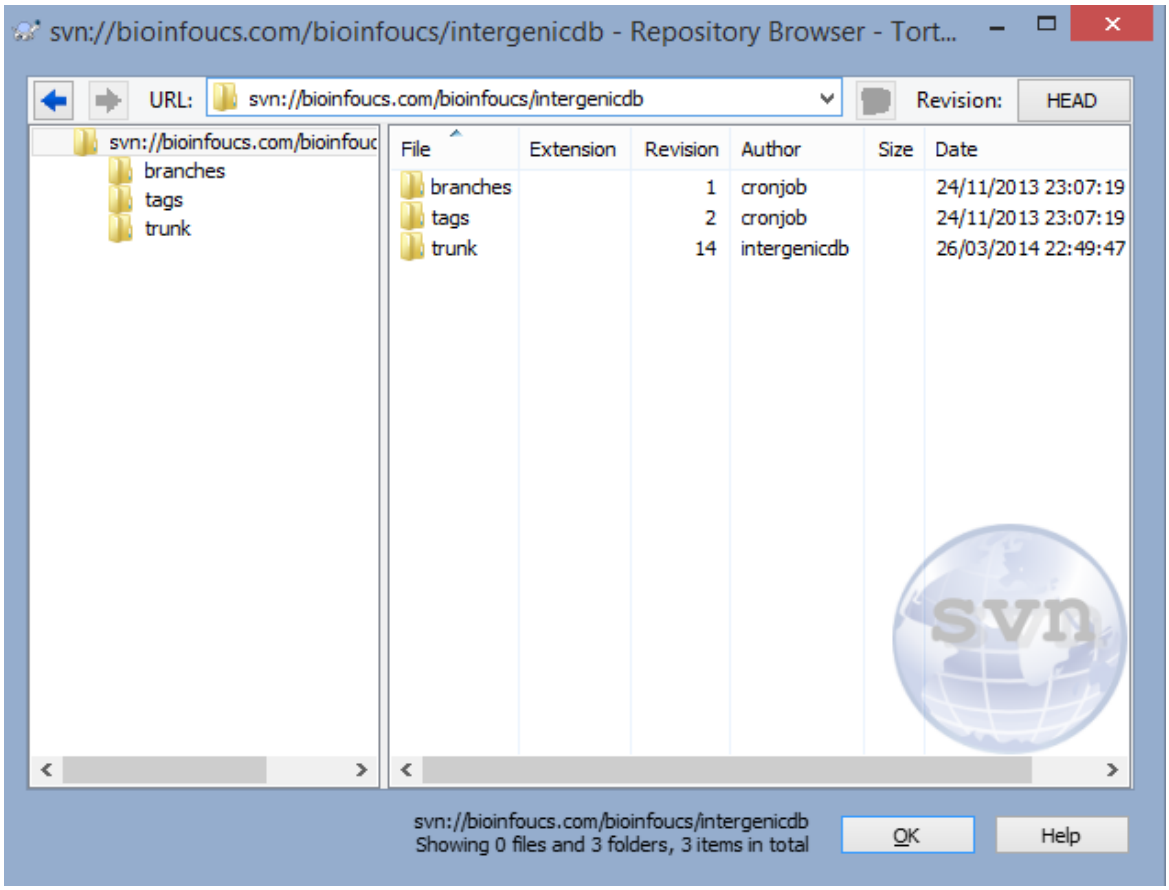

Figura 39 - *Repository Browser* do *TortoiseSVN*

5.2 IMPORTADOR DE DADOS

Para a geração de arquivos que serão utilizados na realização do *upload* no IntergenicDB foi criada uma nova ferramenta denominada MMDBImportTool. A ferramenta MMDBImportTool é composto por cinco projetos(Figura 40).

[25] http://tortoisesvn.tigris.org/

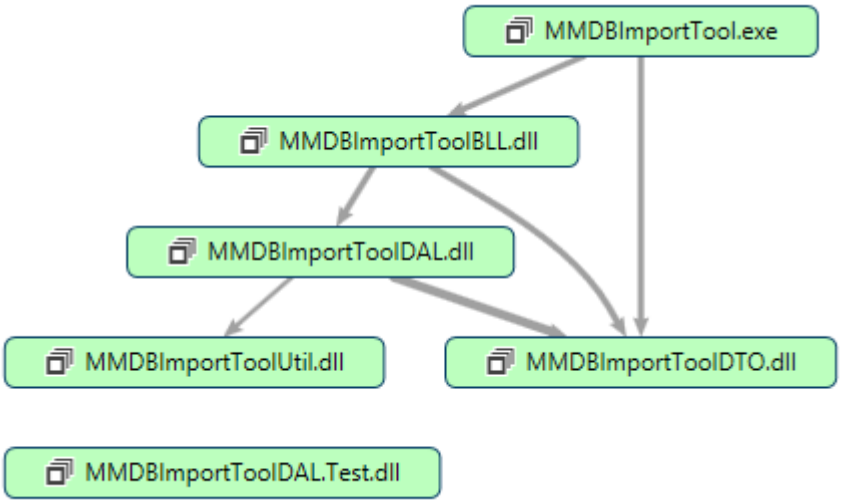

Figura 40 - Estrutura do projeto MMDBImporTool

A visualização fica a cargo do projeto MMDBImportTool. O projeto MMDBImportToolBLL é responsável por implementar as regras do sistema. Já o projeto MMDBImportToolDAL é encarregado de realizar a leitura e escrita de arquivos. O projeto MMDBImportToolDAL possui os modelos de objetos que irão trafegar entre as camadas. Por fim foi criado o projeto MMDBImportToolUtil, responsável por armazenar métodos úteis a toda solução.

Ao entrar no sistema, será exibida para o usuário uma tela com duas abas. Uma para execução do processo de importação de dados e geração do arquivo e outra para configurações. Na aba de configuração é possível definir se o sistema irá utilizar arquivos locais ou arquivos na pasta FTP do NCBI, além de configurações de diretórios para gravação dos arquivos baixados e leitura dos arquivos locais.O sistema não irá buscar nenhuma informação do portal JCVI – CRM, pois durante o processo de desenvolvimento da ferramenta o portal ficou indisponível para acesso aos seus dados.

Na aba para execução do processo são exibidas informações da execução da importação além da possibilidade de definir o organismo a ser importado.

A instalação do sistema pode ser vista no ANEXO A – Processo de instalaçãodo MMDBImporTool. Na execução da importação o sistema irá seguir um fluxo de processo que pode ser visto na Figura 41.

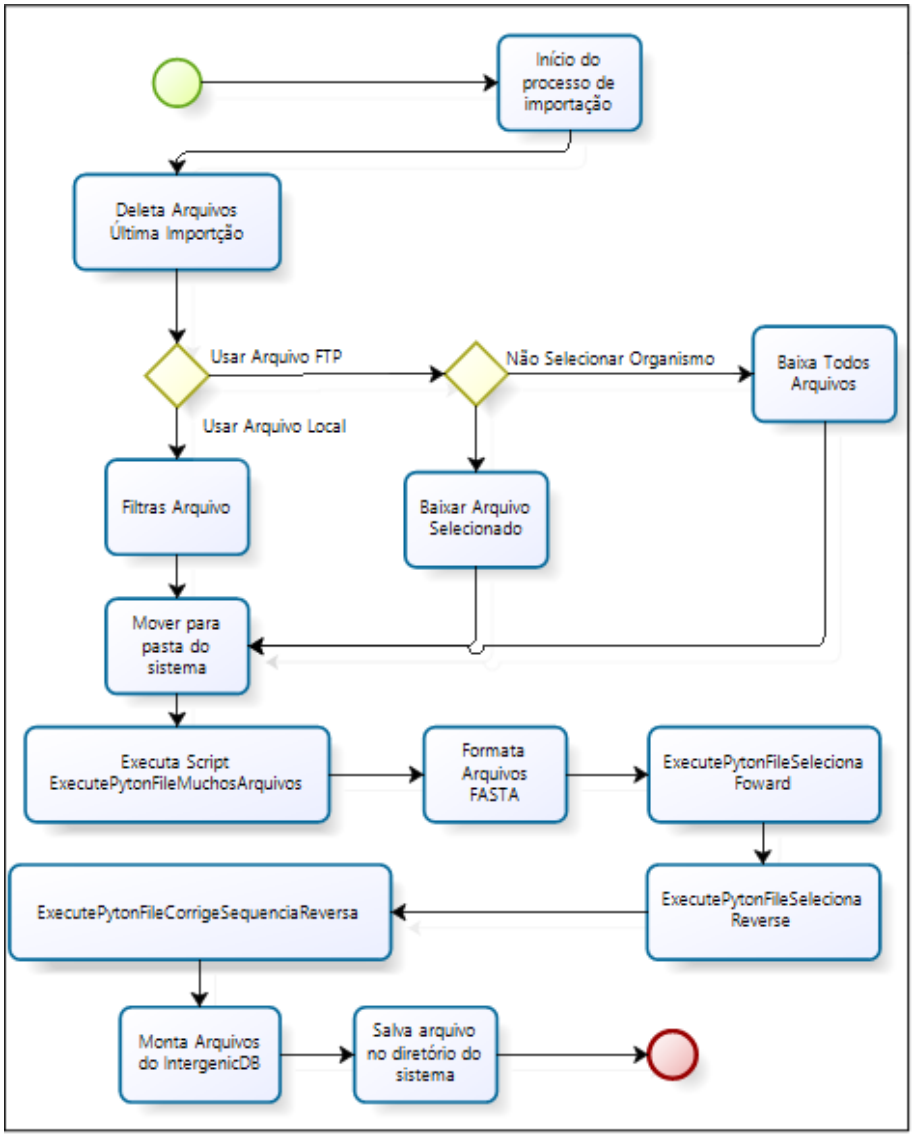

Figura 41 - Fluxo para importação dos dados

5.2.1 *Download* de arquivos

Para o processo de *download* dos arquivos do FTP do NCBI foi criado o método *GetGBKFromFTPNCBI* conforme Figura 42. Este método recebe dois parâmetros, as configurações do sistema (A) e a pasta onde deve ser buscado o arquivo no FTP (B). Dos arquivos existentes na pasta informada, o sistema faz a busca daqueles com extensão *.gbk* (C). Quando um arquivo é encontrado com essa extensão é feito o *download* (D) que será salvo na pasta informada nas configurações passadas por parâmetro para o método (E).

```
A  public void GetGBKFromFTPNCBI(ConfigurationDTO configuratioDTO, string folder)
   {
       string file ="";
B      List<string> files = ListFileFromFTP(configuratioDTO.FTPFolder + folder);
       foreach (string item in files)
       {
C          if (item.ToLower().Contains(".gbk"))
           {
               file = item;
               FtpWebRequest request = (FtpWebRequest)WebRequest.
D                  Create("ftp://ftp.ncbi.nlm.nih.gov/genomes/Bacteria/" + file);
               request.Method = WebRequestMethods.Ftp.DownloadFile;
               request.Credentials = new NetworkCredential("anonymous", "bioinfoucs@gmail.com");
               request.UsePassive = true;
               request.UseBinary = true;
               request.KeepAlive = true;
               FtpWebResponse response = (FtpWebResponse)request.GetResponse();
               Stream responseStream = response.GetResponseStream();
               byte[] buffer = new byte[2048];
               string _diretorio = configuratioDTO.LocalFolder + "\\Arquivos\\" +
                                   file.Replace(file.Substring(item.IndexOf("/")), "");
               if (!Directory.Exists(_diretorio))
               {
                   Directory.CreateDirectory(_diretorio);
               }
               FileStream newFile = new FileStream(_diretorio + "\\" +
                                           file.Substring(item.IndexOf("/") +1), FileMode.Create);
               int readCount = responseStream.Read(buffer, 0, buffer.Length);
               while (readCount > 0)
               {
                   newFile.Write(buffer, 0, readCount);
E                  readCount = responseStream.Read(buffer, 0, buffer.Length);
               }
               newFile.Close();
               responseStream.Close();
               response.Close();
           }
       }
   }
```

Figura 42 - *Download* de arquivos do NCBI

5.2.2 Mover para pasta do sistema

Uma das ações tomadas pelo sistema, não interessando se o arquivo veio do FTP do NCBI ou se já estava armazenado localmente, é mover os arquivos para uma pasta própria um nível abaixo da pasta onde o MMDBImportTool se encontra. Essa ação é feita como garantia para que os arquivos não sejam removidos ou alterados durante o processo de execução.

5.2.3 Execução dos Scripts e Formatação dos arquivos Fasta

O script *ExecutePytonFileMuchosArquivos* é responsável por gerar um arquivo do tipo FASTA que será utilizado pelos outros scripts, porém o arquivo Fasta

é gerado com uma formatação não reconhecida pelos scripts *ExecutePytonFileSelecionaFoward e ExecutePytonFileSelecionaReverse*. Para não alterar os *scripts* em *python* foi criado um método chamado *FormatFasta* que se encarrega de deixar os arquivos FASTA formatos corretamente antes de executar os próximos *sripts*.

5.2.4 Monta Arquivos do IntergenicDB

O principal método criado é denominado *CreateIntergenicDBFiles* é responsável por ler as informações dos arquivos de origem e montá-los no formato correto, levanto em consideração as diferenças existentes entre as sequências intergênicas do tipo *Forward* e *Reverse.*

Os arquivos *Forward* e *Reverse* devem ser tratados de forma diferente, pois como pode ser visto na Figura 43. A forma como deve ser recuperada a informação da posição de início da sequência gênica muda conforme o tipo da sequência.

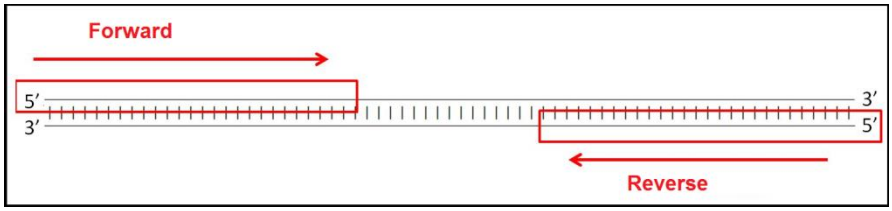

Figura 43 - Leitura de fita foward X fita *reverse*

Dada essa diferença de tratamento foram criados dois métodos distintos para ler as informações dos arquivos, para o *forward* foi criado o método *CreateIntergenicDBFileGBKFoward* e para o *reverse* foi criado o método CreateIntergenicDBFileGBKReverse. Os dois métodos trabalham da mesma forma, com uma diferença na forma de se buscar o início da sequência gênica.

O método *CreateIntergenicDBFileGBKFoward* assume que a sequência gênica inicia na posição seguinte a última posição do sequência intergênica conforme Figura 44.

```
else if (s.IndexOf("     gene            " + (long.Parse(lineDTO.EndSequenciaIntergenic) + 1)) >= 0 ||
           s.IndexOf("     gene              complement(" + (long.Parse(lineDTO.EndSequenciaIntergenic) + 1)) >= 0)
{
```

Figura 44 - Detalhe do método *CreateIntergenicDBFileGBKFoward*

Já o método *CreateIntergenicDBFileGBKReverse* assume que a sequência gênica inicia na posição anterior a última posição do sequência intergênica conforme Figura 45.

```
else if (s.IndexOf("     gene            ") >= 0 &&
           s.IndexOf(".." + (long.Parse(lineDTO.EndSequenciaIntergenic) - 1) + ")") >= 0)
{
```

Figura 45 - Detalhe do método CreateIntergenicDBFileGBKReverse

5.3 TESTES DE CONSULTA DOS DADOS DO IMPORTADOR

Este teste tem por objetivo validar as informações geradas pela ferramenta de importação de dados. Para isso, serão realizadas consultas através da ferramenta de pesquisa disponibilizada pelo portal IntergenicDB.

Para iniciar os testes, foi criada uma base de dados especificamente para isso. Essa base de dados não continha nenhuma informação de regiões intergênicas.

Na ferramenta de importação de dados foram selecionados dois organismos para importação. São eles, *Escherichia coli 536* e *Vibrio cholerae MJ-1236.* O arquivo resultante do processo de importação de dados do NCBI de cada organismo foi importado pelo portal IntergenicDB utilizando seu processo de *upload* de dados.

Genes que possuem *"joins"* na localização da posição nos arquivos do NCBI não foram importados. Foi realizada uma primeira pesquisa com filtro composto utilizando o nome do organismo, o símbolo do gene, o comprimento e a direção da região intergênica para frente (Figura 46).

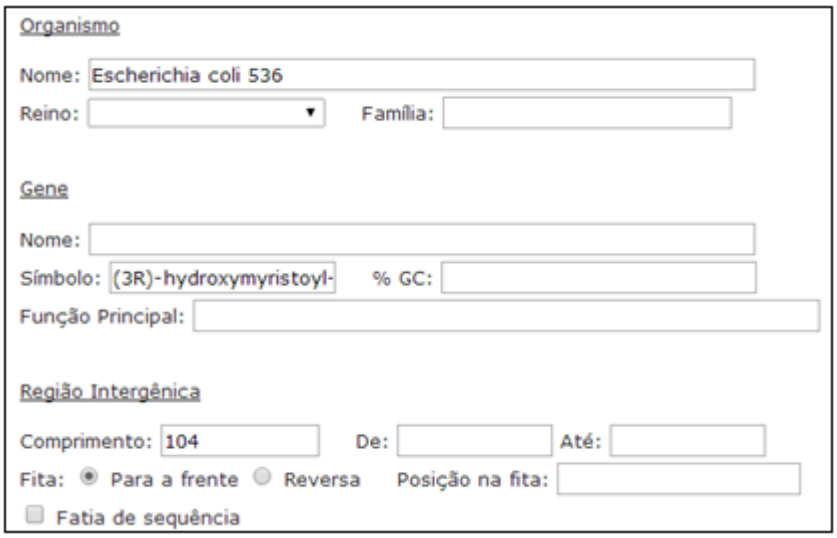

Figura 46 - Condições da pesquisa (Escherichia coli 536)

O resultado da pesquisa foi comparado diretamente com o arquivo baixado pelo FTP do NCBI de forma manual. O resultado da comparação pode ser visualizado na Figura 47, onde é possível visualizar as informações de um dos genes do organismo *Escherichia coli 536*.

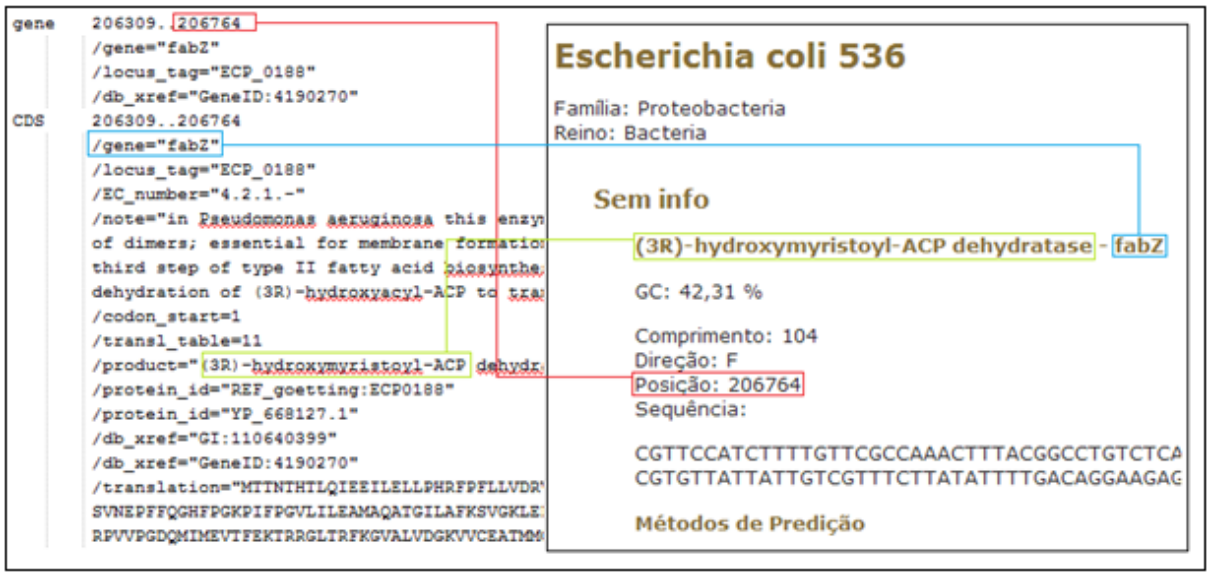

Figura 47 - Comparação do resultado (Escherichia coli 536)

A segunda pesquisa realizada utilizou o organismo *Vibrio cholerae MJ-1236* como parâmetro na pesquisa. Nesta consulta, foram utilizados os filtros de reino do organismo, nome do gene e direção da região intergênica como reversa (Figura 48), direção oposta a utilizada no teste anterior com o organismo *Escherichia coli 536*.

Figura 48 - Condições da pesquisa (Vibrio cholerae MJ-1236)

As informações retornadas pela consulta, assim como no teste anterior, foram comparadas com o arquivo existente no FTP o NCBI. Na Figura 49 foi feita a comparação entre os dados existentes no arquivo com o resultado da pesquisa.

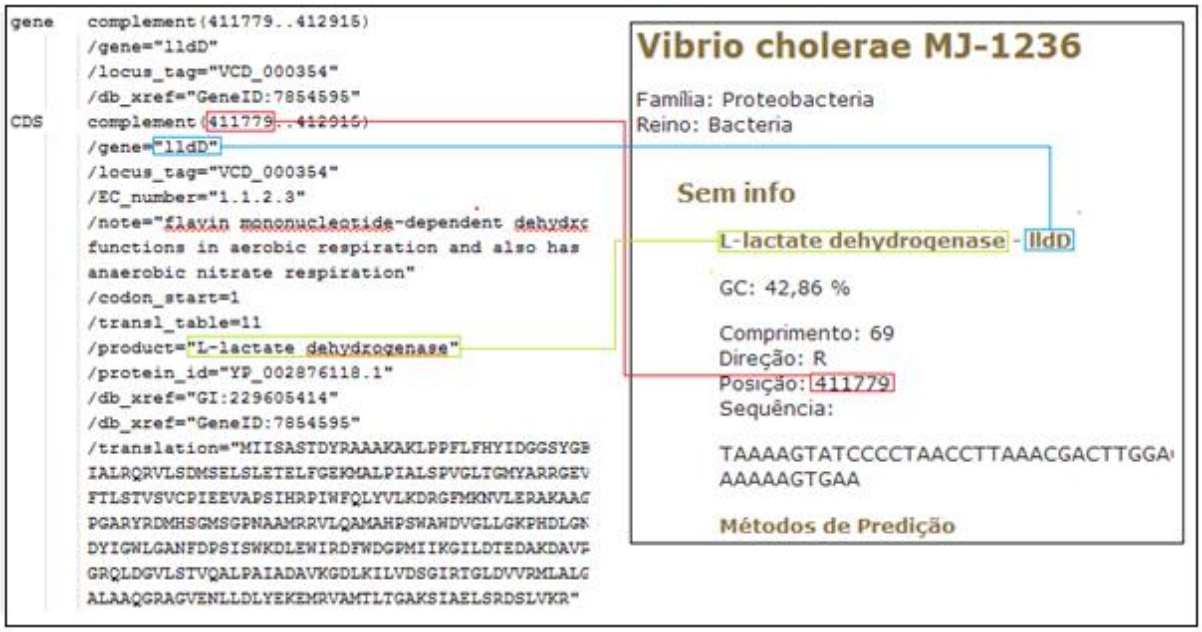

Figura 49 - Comparação do resultado (Vibrio cholerae MJ-1236)

Com os resultados obtidos através dos dois testes executados, é possível observar que as informações geradas pela ferramenta de importação desenvolvida

neste trabalho, que posteriormente foram carregadas para o portal IntergenicDB estão em acordo com os dados existentes nos arquivos onde os dados foram extraídos.

6 CONCLUSÃO

O primeiro passo para o desenvolvimento deste trabalho foi realizar um estudo sobre as ciências envolvidas no projeto. Foi realizado um estudo sobre biologia molecular, com o objetivo de melhor compreender seus conceitos básicos, que associado a pesquisas sobre bioinformática, facilitou o entendimento do portal IntergenicDB e seu objetivo. O portal já estava em desenvolvimento no início deste trabalho, portanto foi necessário ler os trabalhos desenvolvidos até então, para auxiliar na compreensão das funcionalidades do IntergenicDB e entender melhor suas necessidades.

A partir das necessidades apresentadas pelo grupo de Bioinformática da UCS, foram levantados alguns novos requisitos que foram implementados no portal IntergenicDB. A maior necessidade apresentada era a criação de uma ferramenta de importação de dados de outros portais de bioinformática. Os portais escolhidos foram o NCBI e o JCVI - CMR.

Antes da fase de implantação, foi necessário realizar um estudo voltado as ferramentas utilizadas no desenvolvimento do portal e ao próprio código fonte. Um entendimento do *framework* utilizado no IntergenicDB era necessário antes de começar a etapa de implantação.

Na fase de implementação dos requisitos no portal surgiram alguns imprevistos. A *procedure* que grava as informações temporárias em tabelas apropriadas não estava controlando informações duplicadas. Durante o desenvolvimento do projeto, o serviço de hospedagem onde o portal é publicado alterou suas diretivas de segurança deixando o IntergenicDB inutilizável, consumindo tempo e esforço para resolver esse problema.

Porém as maiores dificuldades foram encontradas no desenvolvimento da ferramenta de importação de dados. No início do desenvolvimento foi constatado que portal JCVI – CRM não disponibiliza mais acesso aos seus dados. Além criação de um arquivo com informações originárias da junção de dados de mais de um arquivo também se mostraram um desafio.

Superados os problemas, ao final do projeto, os requisitos foram implementados com sucesso. Agora o portal conta com uma ferramenta de recuperação de senha, possui a informação do país no cadastro de usuário, é capaz

de importar mais de um arquivo por vez. As mudanças realizadas no cadastro de genes, que resultaram na criação de novas tabelas no banco de dados e novas áreas de manutenção no portal tornaram o IntergenicDB mais robusto.

Na implantação da nova ferramenta de importação de dados, além de realizar o que foi proposto inicialmente, foram desenvolvidas algumas melhorias para facilitar o seu uso. A principal melhoria está na possibilidade de trabalhar com informações que não estejam no FTP do NCBI, deixando para o usuário escolhar a forma como a ferramenta irá trabalhar, tornando o *software* mais flexível no seu uso.

Ambas as ferramentas tem potencial para fornecer novas funcionalidade e se tornarem ferramentas cada vez mais completas.

6.1 TRABALHOS FUTUROS

Entre os trabalhos que podem ser realizados futuramente estão a reescrita dos *scripts* hoje utilizados na ferramenta de importação de *python* para *C#*, deixando integrado ao importador de dados. Além disso, pode-se pensar em começar a trabalhar com um banco de dados específico para a nova ferramenta, tornando-o mais robusto e não deixando mais que ele trabalhe apenas com leitura e escrita de arquivos texto.

7 REFERÊNCIAS BIBLIOGRÁFICAS

ALBERTS, B.; BRAY, D.; LEWIS, J.; RAFF, M.; ROBERTS, K.; WATSON, P. D. Biologia Molecular da Célula. ArtesMédicas, 1997.

ALBERTS, B.; BRAY, D.; JOHNSON, A.; LEWIS, J.; RAFF, M.; ROBERTS, K.; WALTER, P. Fundamentos da biologia celular. ArtMed, 2011.

ÁVILA E SILVA, S. de, Redes neurais artificiais aplicadas no reconhecimento de regiões promotoras em bactérias Gram-negativas. 2011. 82 f. Tese (Doutorado) - Curso de Biotecnologia, Universidade de Caxias do Sul, Caxias do Sul, 2011.

BACK, K. TDD Desenvolvimento Guiado por Teste. Bookman, 2010.

CASTRAVECHI , D.; Shinoda, A. A. Desenvolvimento de um Sistema para Armazenamento de Seqüências Nucléicas. [Online] II Workshop de Tecn. da Inf. aplicada ao Meio Ambiente – CBComp: 2004. [Acessado em 12 de agosto de 2013]. Disponível na internet em http://www.niee.ufrgs.br/eventos/CBCOMP/2004/pdf/Workshop_Ambiente/Bioinformatica_Sequenciamento_DNA/t170100217_3.pdf

CHAPMAN, B.; CHANG J. Biopython: Python tools for computational biology. [Online] ACM SIGBIO Newsletter, 2000 [Acessado em 11 de junho de 2014]. Disponível na internet em http://biopython.org/static/DIST/docs/acm/ACMbiopy.pdf

COLLINS-SUAAMAN, B.; FITZPATRICK, B. W.; PILATO C. M. Version Control with Subversion For Subversion 1.7. [Online, Acessado em 07 de maio de 2014]. Disponível na internet em ftp://ftp6.tw.freebsd.org/FreeBSD/ports/local-distfiles/lev/svn-book-r4515.pdf

DAVANZO, V. Desenvolvimento de Consultas para um Banco de Dados de Sequências Intergênicas. Centro de Computação e Tecnologia da Informação,

Universidade de Caxias do Sul: Caxias do Sul: 2010. Bacharelado em Ciências da Computação – Trabalho de Conclusão de Curso.

DAVIDSEN, T.; Beck E.; Ganapathy A.; Montgomery R.; Zafar N.; Yang Q.; Madupu R.; Goetz P.; Galinsky K.; White O.; Sutton G. The comprehensive microbial resource. [Online] J. Craig Venter Institute, Rockville, MD 20850 and Institute for Genome Sciences, University of Maryland School of Medicine, Baltimore, MD 21201, USA: 2009. [Acessado em 19 de novembro de 2013]. Disponível na Internet em http://nar.oxfordjournals.org/content/38/suppl_1/D340.full.pdf+html

DIAS CORREIA, J. H. R.; Dias Correia, A. A. Funcionalidades dos RNA não codificantes (ncRNA) e pequenos RNA reguladores, nos mamíferos. [Online] REDVET. Revista Electrónica de Veterinária: 2007. [Acessado em 27 de agosto de 2013]. Disponível na internet em http://www.veterinaria.org/revistas/redvet/n101007/100705.pdf

FOWLER, M.; Rice, D.; Foemmel, M.; Hieatt, E.; Mee, R.; Stafford, R. Patterns of Enterprise Application Architecture. Addison Wesley, 2002.

GAMMA, E.; Helm R.; Johnson, R; Vlissides, J . Design Patterns – Eleme
Reusable Object-Oriented Software. 1 ed. Addison-Wesley Professional, 1994.

JUNQUEIRA, L. C.; Carneiro, J. Biologia Celular e Molecular. 8° Edição. Rio de Janeiro: Guanabara Koogan, 2005.

LESSA, F. A. Identificação de RNAs não-codificadores por modelos de covariância com prioris Dirichlet adaptadas a grupos de ncRNAs com estruturas secundárias similares. [Online] Departamento de Ciência da Computação, Universidade de Brasília: Brasília: 2012.
[Acessado em 14 de agosto de 2013]. Disponível na internet em http://repositorio.unb.br/bitstream/10482/11840/1/2012_FelipeAlmeidaLessa.pdf

MIZRACHI I. GenBank. 2013 Nov 12. In: The NCBI Handbook. 2nd edition. Bethesda (MD): National Center for Biotechnology Information (US); 2013-. [Acessado em 28 de novembro de 2013]. Disponível na internet em http://www.ncbi.nlm.nih.gov/books/NBK153518/

MOLIN, A. F. Prototipagem de um Banco de Dados de Promotores como Base para um Portal de Serviços. Centro de Computação e Tecnologia da Informação, Universidade de Caxias do Sul: Caxias do Sul: 2009. Bacharelado em Ciência da Computação – Trabalho de Conclusão de Curso.

NOTARI, D. L. Desenvolvimento de workflows científicos para a geração e análise de diferentes redes interatomas. 2012. 161 f. Tese (Doutorado) - Curso de Biotecnologia, Universidade de Caxias do Sul, Caxias do Sul, 2012.

PICOLLOTO, D. Implementação de novas funcionalidades para o Portal IntergenicDB. Centro de Computação e Tecnologia da Informação, Universidade de Caxias do Sul: Caxias do Sul: 2012. Bacharelado em Sistemas de Informação – Trabalho de Conclusão de Curso.

SEIBEL, L. F. B.; Lemos, M.; Lifschitz, S. Bancos de Dados de Genoma. [Online] Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro: Rio de Janeiro: 2000. [Acessado em 12 de agosto de 2013]. Disponível na Internet em http://www.uniriotec.br/~seibel/Tutorial%20SBBD2000%20FinalRev.pdf

SMITH K. A Brief History of NCBI's Formation and Growth. In: The NCBI Handbook. 2nd edition. Bethesda (MD): National Center for Biotechnology Information (US); 2013-. [Acessado em 28 de novembro de 2013]. Disponível na Internet em http://www.ncbi.nlm.nih.gov/books/NBK148949/

ZAHA, A. Biologia Molecular Básica. Porto Alegre: Mercado Aberto, 1996.

ANEXO A – Processo de instalaçãodo MMDBImporTool

Neste anexo será demonstrado o processo de instalação do MMDBImportTool em uma máquina com Windows 8.

Instalação do Python

Para que seja possível a execução de scripts Python pelo MMDBImportTool, é necessário que esteja instalada as ferramentas necessárias para executar os scripts escritos em Python. No nosso caso, é necessário instalar um interpretador de código.

Deverá ser instalada a versão 3.3[26] para que seja mantida a compatibilidade com as bibliotecas utilizadas pelos scripts que serão executados no processo de importação de dados.

Ao executar o instalador será apresentada uma tela para que seja escolhido o local de instalação da aplicação. Em seguida é apresentada uma tela que possibilita escolher os componentes que serão instalados. Nesse momento é importante que a opção *"Add python .exe to Path"* seja habilitada para instalação (Figura 50).

Figura 50 - Instalação do Python 3.3

[26] https://www.python.org/download/releases/3.3.0/

Caso a opção não seja habilitada ou já exista uma instalação do Python, é necessário garantir que o diretório de instalação esteja informado na variável de sistema "*Path*".

As variáveis de sistema podem ser acessadas através da tela de "Propriedades do Sistema" que se encontra nas configurações avançadas do sistema conforme pode ser visto na figura (Figura 51).

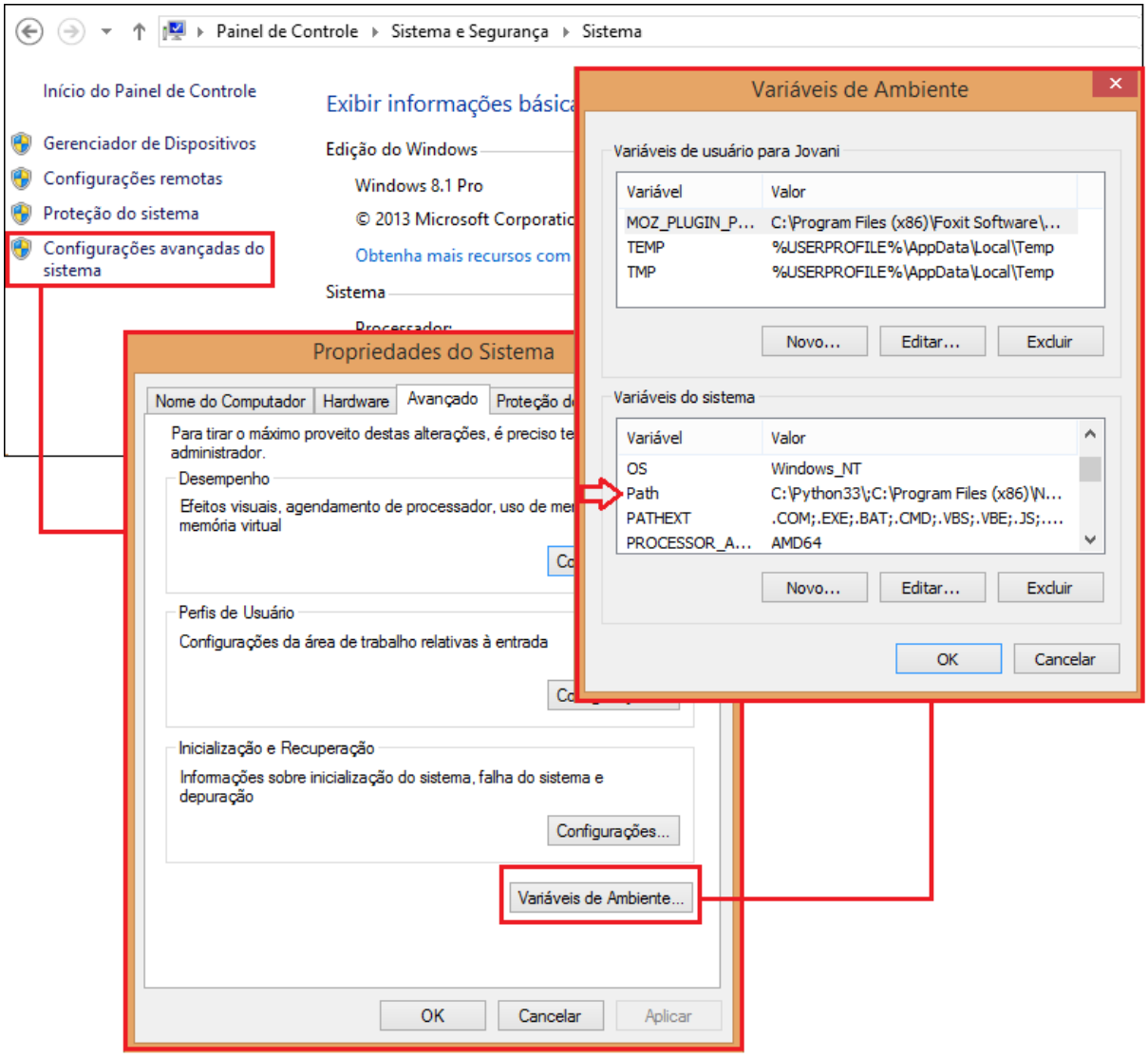

Figura 51 - Acesso a variáveis do sistema

Instalação do BioPython

BioPython[27] é uma ferramenta de código aberto escrita em python criada em agosto de 1999 que tem como objetivo servir de como central de códigos de ferramentas para bioinformática que pesquisadores possam fazer uso(CHAPMAN e CHANG, 2000).

A instalação é realizada através do instalador que pode ser obtido no *site*[28] do BuiPython. É necessário apenas seguir os passos exibidos pelo instalador em tela, o único requisito que deve ser observado é a versão do BioPython deve ser compatível com o interpretado instalado anteriormente. Para o nosso cenário, deve ser baixada versão do BioPytho 1.64[29] compatível com o Python 3.3.

Instalação do MMDBImportTool

A instalação do MMDBImportTool se resume a manter os seguintes arquivos no mesmo diretório:

- MMDBImportTool.exe
- MMDBImportTool.exe.manifest
- MMDBImportToolBLL.dll
- MMDBImportToolDAL.dll
- MMDBImportToolDTO.dll
- MMDBImportToolUtil.dll
- 01_intergenic_muchos_archivos.py
- 02_seleciona_foward_intergenic_bac.py
- 02_seleciona_reverse_intergenic_bac.py
- 03_corrigeSequenciaReversa_bac2.py

[27] http://biopython.org/
[28] http://biopython.org/wiki/Download
[29] http://biopython.org/DIST/biopython-1.64.win32-py3.3.exe