# Software para análise de variantes genômicas nos genes BRCA1 e BRCA2 a partir de dados obtidos em NGS

**Mário M. T. B. de Rezende**

# Software para análise de variantes genômicas nos genes BRCA1 e BRCA2 a partir de dados obtidos em NGS

**Mário M. T. B. de Rezende**

*Supervisor:* Euclides Matheucci Jr.
*Empresa:* QGene Ind. Com. de Equipamentos Para Laboratórios. Ltda.

Monografia de conclusão de curso apresentada ao Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo para obtenção do título de Bacharel em Ciências de Computação.

**USP - São Carlos**
**Maio de 2014**

# Dedicatória

Dedico minha monografia aos meus pais, que sempre me apoiaram a perseguir meus sonhos.

# Agradecimentos

Agradeço a todas as pessoas que fizeram parte de minha vida até o presente momento.

# Resumo

O objetivo deste trabalho é desenvolver um sistema em forma de pipeline para analisar dados produzidos por *Next Generation Sequencing* (NGS). NGS é uma tecnologia recente para sequenciamento de DNA, que produz dados que podem ser analisados para a emissão de um prognóstico sobre o paciente. O sistema visa auxiliar os profissionais da empresa a analisar mutações e emitir o laudo. O sistema está sendo desenvolvido utilizando-se ferramentas e metodologias que permitem fácil prototipação.

**Palavras-chaves:** rails, TDD, bioinformática, DNA.

# Sumário

# Lista de Listagens

Capítulo

# 1

# Introdução

## 1.1 Sobre a Empresa

A QGene Ind. Com. de Equipamentos Para Laboratórios. Ltda. foi fundada em 21 de julho de 2006 e instalada na incubadora CEDIN (Centro de Desenvolvimento de Industrias Nascentes), do sistema de incubadoras da FIESP. A partir de 2009, a empresa foi transferida para nova sede situada na Rua Santa Cruz, 969, em São Carlos, para comportar o crescimento da empresa. A QGene surgiu como uma *spin-off* da DNA Consult Genética e Biotecnologia Ltda., empresa fundada há 12 anos e especializada em análises do DNA, com forte atuação em P&D (possui em seu portfólio três projetos PIPE-FAPESP). Entre janeiro e dezembro de 2007, a QGene abrigou um projeto PIPE-FAPESP relacionado com monitoramento e diagnóstico de vírus em camarões cultivados, tendo recebido cerca de R$ 150.000,00 da FAPESP, para compra de insumos, equipamentos e bolsas.

A QGene atua na área de genética molecular, dedicando-se ao P&D, produzindo e comercializando *kits* para análise de DNA e soluções em saúde humana, que representem inovações tecnológicas e que substituem a importação de produtos.

A QGene é uma empresa de pequeno porte que goza de parcerias com diversas empresas, como a DNA Consult e MD Genetics. Por ser uma empresa de pequeno porte, a QGene também está sempre procurando e atuando em nichos que não eram o foco inicial da empresa, além de seus produtos consolidados há anos no mercado. Atualmente, a QGene está produzindo soluções de bioinformáticas com foco clínico, uma vez que os softwares existentes, apesar de robustos, não apresentam facilidade de uso para médicos e muitas vezes são softwares abrangentes que não apresentam soluções clínicas.

A empresa valoriza muito seus funcionários e não mede gastos para aprimorá-los cada vez mais. A empresa entende que os funcionários, sejam os contratados como os estagiários, podem ajudar a QGene a estar sempre se reinventando como empresa e atacando novos nichos ou setores do mercado.

## 1.2 Sobre o Processo Seletivo

O processo seletivo por qual passei foi apenas uma entrevista. Fui entrevistado por dois sócios da empresa. Um deles possui doutorado na área de biologia molecular, e o outro sócio trabalhava no Banco do Brasil, no cargo de presidência, mas agora está aposentado. A entrevista não continha perguntas técnicas sobre computação, especialmente pois a QGene não possuia, até então, um setor de TI. O que os entrevistadores avaliaram foi algo mais subjetivo e de interesse deles: capacidade de trabalho em equipe, interesse em trabalhar com uma área interdisciplinar (no caso, bioinformática), capacidade de trabalhar de forma autônoma (ou seja, sem uma pessoa de TI me liderando), e proatividade.

## 1.3 Apresentação da monografia

Os próximos capítulos irão retratar de forma mais detalhada as atividades realizadas por mim durante o estágio.

No Capítulo 2, descrevo brevemente como foi o planejamento das atividades. É apresentado também como foi meu treinamento.

No Capítulo 3, apresento algumas ferramentas que utilizo para desenvolver o sistema. Também explico brevemente os motivos que me levaram a adotar a metodologia de *Test Driven Development* (TDD).

No último capítulo, são levantadas as conclusões sobre o estágio e a relação com o curso de graduação, indicando melhorias e perspectivas para o futuro, tanto para o curso quanto para o trabalho.

Capítulo

# 2

# Planejamento do trabalho

## 2.1 Atividades planejadas para o estágio

A atividade planejada é a implementação de um *pipeline* para análise de dados de NGS e gerar laudos automaticamente. Os equipamentos de NGS [3] produzem *output* em um formato padrão que pode facilmente passar por um processo de *parsing* por outras ferramentas. O objetivo deste sistema é que ele faça processamentos de tal *output* com o intuito de melhorar a confiabilidade do sequenciamento (eliminar falsos negativos, especialmente) e melhorar a vizualização dos dados para ajudar o usuário a tomar decisões pertinentes sobre a emissão do laudo.

Como a bioinformática voltada para análises genéticas é uma área nova para mim, o desenvolvimento do sistema computacional não contou com um cronograma. Ao invés disso, o desenvolvimento e o estudo sobre tal área da bioinformática é realizado de forma altamente dinâmica, à medida que surge a necessidade.

A única atividade que foi planejada foi a minha participação no curso de vereão de bioinformática ministrado na Faculdade de Medicina de Ribeirão Preto (FMRP) [1]. O curso foi bem proveitoso e pude melhorar em muito minha comunicação com a equipe da empresa e meu domínio de tal área.

## 2.2 Treinamentos planejados para o estágio

O único treinamento planejado foi a participação no curso da FMRP. Demais treinamentos são realizados quando se fazem necessários.

Capítulo

# 3

# Desenvolvimento do trabalho

## 3.1 Atividades realizadas

A atividade realizada foi a implementação do pipeline para análise de dados de NGS e construção automática de laudo. Por ser um sistema complexo e, pela falta de experiência do aluno na área de genética, até o momento foram implementados apenas alguns protótipos que estão servindo para melhorar a comunicação com profissionais da área clínica e para ajudar a levantar os requisitos do sistema.

A metodologia utilizada para a construção dos protótipos é TDD.

## 3.2 Problemas resolvidos

Os dados gerados pelo NGS [2] precisam ser processados antes de serem analisados. Até o presente momento, os protótipos desenvolvidos ajudaram a evidenciar alguns erros que estavam sendo cometidos no processamento de tais dados e me permitiram estudar de forma mais concreta tal tecnologia.

## 3.3 Técnicas, métodos e tecnologias envolvidas

Aqui, são apresentados alguns detalhes sobre o desenvolvimento do sistema. As ferramentas usadas no desenvolvimento são bem documentadas e apresentam uma abundância de exemplos na internet. A descrição detalhada de cada ferramenta está fora do escopo deste trabalho.

### 3.3.1 Framework

O framework usado para a implementação do sistema é Ruby on Rails (RoR) [5]. RoR é um framework poderoso para construção de aplicativos web. RoR implementa de forma simples e elegante o modelo *Model-view-controller* (MVC).

RoR foi escolhido por alguns motivos. É um framework com alto poder de prototipação, abstrai totalmente a parte de banco de dados (SQL só é necessário para otimizar certas consultas), conta com uma comunidade muito ativa, pode ser facilmente configurado para utilizar os serviços de *storage* da Amazon, e é um framework que incentiva o programador a escrever casos de teste a fim de ter maior segurança durante a codificação e refatoração do código.

### 3.3.2 RSpec

RSpec [6] é uma biblioteca (no contexto de RoR, também chamadas de *gemas*) projetada para a descrição e implementação de **casos de teste** (CT). A sintaxe do RSpec é muito parecida com a sintaxe da linguagem Ruby ao mesmo tempo que se parece muito com a língua inglesa (como exemplificado na Listagem 3.1). Isso faz com que os CT sejam muito descritivos e concisos, tanto para o desenvolvedor quanto para o cliente, diminuindo a dificuldade de comunicação.

É possível escrever CT para os modelos, controladores, e até mesmo para as views do sistema. Com a disponibilidade de uma gema tão robusta e bem projetada, tornou-se uma convenção a utilização de TDD no desenvolvimento de sistemas escritos em RoR.

**Listing 3.1:** Exemplo de RSpec (sem outras gemas)

```
1
2 describe Pdf do
3         specify "when title is not unique" do
4                 pdf.should_not be_a_new_record
5                 expect { Pdf.create!(pdf.attributes) }.to raise_error
                      ActiveRecord::RecordInvalid
6         end
7 end
```

### 3.3.3 Shoulda e Capybara

Shoulda [7] e Capybara [4] são gemas que incrementam a funcionalidade do RSpec.

Shoulda possui métodos para testar controladores ou modelos com apenas uma linha de código. Por exemplo, existe um método (Listagem 3.2) que verifica se a tabela correspondente a um modelo foi criada no banco de dados, se os campos da tabela estão corretos, e se as chaves estrangeiras estão referenciando as tabelas corretas. Essa gema não foi utilizada no início do

estágio, mas quando foi adotada no projeto, o tamanho dos arquivos de CT foi reduzido consideravelmente. Isso foi muito importante, pois os protótipos não contam com um documento de requisitos, apenas com os CT, e pode-se configurar o Rspec para que ele solte um output em formato de documentação. Deixar a implementação dos CT concisa reduziu em muito o tamanho da documentação gerada pelo Rspec.

**Listing 3.2:** Exemplo de Shoulda

```
1
2 describe Post do
3         it { should belong_to :user }
4 end
```

Capybara é uma gema com métodos para se realizar testes de integração. Testes de integração são testes que simulam um usuário utilizando o sistema para determinadas tarefas. Por exemplo, alguns métodos da gema (Listagem 3.3) permitem simular um usuário que visita uma página, preenche um formulário, e pressiona o botão de submissão do formulário. Se, após tudo isso, o usuário for redirecionado para a página correta e os dados forem salvos no banco de dados, então o aplicativo passa nesse caso de teste.

**Listing 3.3:** Exemplo de Capybara

```
1
2 describe "signup" do
3         visit signup_path
4         fill_in "Name", with: "Example User"
5         fill_in "Email", with: "user@example.com"
6         fill_in "Password", with: "foobar"
7         fill_in "Confirmation", with: "foobar"
8
9         # expectations ...
10 end
```

### 3.3.4 TDD

TDD é uma metodologia de desenvolvimento onde os CT são escritos antes da aplicação ser implementada. Tais casos de teste guiam o processo de desenvolvimento. Certos CT são escritos consultando-se o cliente, de tal forma que os casos de teste possam ser vistos como uma forma de documento de requisitos do sistema.

Além de guiar o desenvolvedor a implementar exatamente o que o cliente deseja, os casos de teste evitam possíveis *bugs*. Esse aspecto é extremamente importante para aplicações em RoR, pois tal framework é uma extensão da linguagem Ruby, uma linguagem com sintaxe e semântica bastante flexissíveis e permissivas. RoR também conta com algumas outras ferramentas para o

funcionamento do sistema, como Javascript, SQL, HTML, e CSS. Um sistema escrito em RoR que não possui casos de teste robustos geralmente é um sistema muito frágil, especialmente quando manutenções forem necessárias. Assim, TDD é uma metologia que é usada por grande parte da comunidade que utiliza RoR.

## 3.4 Impacto

Quando o sistema ficar pronto, os profissionais terão em mãos uma ferramente valiosa para ajudá-los a analisar os dados do sequenciamento. Usando a ferramenta, espera-se que os prognósticos sejam mais precisos e possam ser usados para previnir casos clínicos antes que estes ocorram.

A empresa está firmando acordos com algumas entedidades, porém tais informações são sigilosas.

## 3.5 Problemas não resolvidos

Não existe versão final do sistema ainda. Apenas protótipos que estão sendo usados para explorar possíveis funcionalidades e outros aspectos do problema.

Capítulo

# 4

# Conclusão

## 4.1 Benefícios para o crescimento profissional

A oportunidade de trabalhar sem ser liderado por alguém da área de TI me interessou muito, pois vi isso como uma forma de aprender muitas coisas, desde liderança (atualmente, a empresa contratou um outro estagiário para que me ajude e seja meu subordinado), até coisas mais específicas da computação, como a parte de conversar com o cliente, fazer o levantamento dos requisitos do sistema, e tomar decisões sobre linguagens de programação, metologia de desenvolvimento e muitas outras coisas.

Outro aspecto interessante e útil de estar trabalhando na empresa é a capacidade de aumentar meu *networking* (rede de contatos). Estou envolvido diariamente com diversas pessoas, desde economistas, biólogos, médicos, publicitários, e até mesmo, como mencionado anteriormente, com o ex-presidente do Banco do Brasil. Tal *networking* diversificado me ajuda cada vez mais a aprimorar meu perfil de empreendedor e acredito que seja uma excelente forma de treinar a habilidade de definir produtos com futuros clientes e trabalhar em áreas interdisciplinares.

## 4.2 Considerações sobre o curso de graduação

O curso forneceu as bases teóricas para que o estágio fosse desenvolvido. Ainda que minha experiência no mercado de trabalho seja algo recente, as bases teóricas me facilitaram bastante a aquisição de novos conhecimentos com finalidade de elevar o meu trabalho desenvolvido durante o estágio ao nível profissional.

## 4.3 Sugestões para o curso de graduação

As disciplinas de engenharia de software que cursei não abordavam TDD. Acredito que esta metodologia de desenvolvimento seja muito importante e interessante, e deveria ser abordada.

Criptografia é um assunto pouco abordado no curso. Estudei criptografia apenas nas matérias de redes. Durante o estágio, foi necessário utilizar criptografia em outras camadas da aplicação, como, por exemplo, no banco de dados. Acredito que criptografia seja um assunto que deva ser abordado também nas disciplinas de bancos de dados.

As disciplinas de matemática não foram muito proveitosas. Não usei cálculo nem geometria analítica, por exemplo. Acredito que tais disciplinas ainda sejam muito importantes, pois fornecem bases lógicas e são requisitos para outras disciplinas. Mas a carga de matemática do curso, em minha opinião, deveria incluir mais disciplinas de matemática que possam ser usadas em computação diretamente do que disciplinas que são usadas em contextos mais específicos. Considero que cálculo seja uma disciplinas de contexto específico, e disciplinas como matemática discreta e probabilidade sejam mais aplicáveis em áreas diversas da computação. Tive a oportunidade de cursar outras disciplinas do curso de matemática, fornecidas como optativas para a computação. Algumas delas foram Álgebra 1, Elementos de Matemática, Tópicos de Matemática Elementar, e Tópicos de Otimização Combinatória. O conteúdo de tais disciplinas é também fornecido no curso de computação, porém de forma muito mais resumida (nas disciplinas de matemática discreta). Acredito que as disciplinas de discreta não deveriam ser tão condensadas/resumidas.

Outra disciplinas que deveria receber maior atenção é a de empreendedores em informática. Considero que a linha divisória entre um funcionário e um empreendedor seja muito subjetiva. Eu, como funcionário, muitas vezes recorro a conceitos vistos na disciplinas de empreendedorismo. Tal disciplinas deveria ser obrigatória no curso.

As disciplinas preparatórias para a maratona de programação também são muito importantes. Pude ganhar mais experiência na parte de modelar o problema, testar, implementar e otimizar a implementação para que o programa seja executado de forma rápida. Nessas disciplinas, também melhorei minha habilidade de trabalhar em equipe, pois o time para a maratona é composto por três pessoas. Muitos dos conceitos absorvidos em tais disciplinas não são aplicados extensivamente, pois a prioridade no trabalho sendo desenvolvido é manutenibilidade do código, ao invés de código extremamente eficiente. Porém, tais disciplinas me deram uma melhor capacidade de resolver os problemas que podem surgir, e acredito que foram as responsáveis por interconectar as matérias teóricas com as matérias práticas.

A única reclamação que tenho sobre o curso é sobre a duração das aulas. A maioria das aulas tem duração de duas horas. Isso já é uma duração longa, e outras aulas podem ter até quatro horas de duração. É um contraste muito grande com as aulas do MIT, por exemplo, que tem duração de no máximo uma hora.

Em suma, estou satisfeito com o curso. Porém, tal satisfação só foi atingida porque busquei estudar coisas extracurriculares. Infelizmente, não pude ver isso em muitos colegas, que se formaram antes de mim e que ainda assim sentiram que o curso não foi proveitoso. Acredito que seja possível reestruturar o curso de tal forma que futuros formandos entrem mais confiantes no mercado de trabalho.

## 4.4 Planos para o futuro

Pretendo continuar trabalhando na empresa. Acredito que seja um lugar onde eu possa adquirir muita experiência, principalmente em como trabalhar conjuntamente com profissionais de outras áreas.

# Referências


[1] FMRP. 10th Summer Bioinformatics Course. http://bioinformatics.fmrp.usp.br/curso2014/. Acessado: 2014-05-28.

[2] 1000 Genomes. SAM/BAM and related specifications. https://github.com/samtools/hts-specs. Acessado: 2014-05-28.

[3] Illumina. Next-Generation Sequencing Technology. http://www.illumina.com/technology/next-generation-sequencing.ilmn. Acessado: 2014-05-28.

[4] Jonas Nicklas. Capybara GitHub page. https://github.com/jnicklas/capybara. Acessado: 2014-05-28.

[5] Ruby on Rails. Ruby on Rails GitHub page. https://github.com/rails/rails/. Acessado: 2014-05-28.

[6] RSpec. RSpec GitHub page. https://github.com/rspec/rspec-rails/. Acessado: 2014-05-28.

[7] Thoughtbot. Shoulda GitHub page. https://github.com/thoughtbot/shoulda/. Acessado: 2014-05-28.