INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
CAMPUS CAJAZEIRAS
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

RELATO DE EXPERIÊNCIA NO MERCADO DE TRABALHO: NITRYX CONSULTING

MICHELLE OLIVEIRA DA COSTA

CAJAZEIRAS-PB
2021

MICHELLE OLIVEIRA DA COSTA

RELATO DE EXPERIÊNCIA NO MERCADO DE TRABALHO: NITRYX CONSULTING

Trabalho de Conclusão de Curso apresentado junto ao Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia da Paraíba - Campus Cajazeiras, como requisito à obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador

Prof. Me. Diogo Dantas Moreira

CAJAZEIRAS

2021

Dados Internacionais de Catalogação na Publicação (CIP)

C837r Costa, Michelle Oliveira da

Relato de experiência no mercado de trabalho: Nitryx Consulting/Michelle Oliveira da Costa. – Cajazeiras/PB: IFPB, 2021.

36f.: il.

Trabalho de Conclusão de Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas - Instituto Federal de Educação, Ciência e Tecnologia da Paraíba-IFPB, Campus Cajazeiras. Cajazeiras, 2021.

Orientador(a): Prof. Me. Diogo Dantas Moreira.

1. Automação - Desenvolvimento de sistemas 2. Automação – Nitro MP 3. Automação – Nitro Yard 4. Nitryx Consulting I. Costa, Michelle Oliveira da II. Título

CDU: 004.45

Ficha catalográfica elaborada pelo Departamento de Bibliotecas DBIBLIO/IFPB

SERVIÇO PÚBLICO FEDERAL
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
COORDENAÇÃO DO CURSO SUPERIOR EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - CAMPUS CAJAZEIRAS

ATA DE DEFESA DO TRABALHO DE CONCLUSÃO DE CURSO (TCC)
CURSO: ANÁLISE E DESENVOLVIMENTO DE SISTEMAS (ADS)

Às 15h00 do dia 20 do mês de DEZEMBRO do ano de 2021, o(a) aluno(a) **MICHELLE OLIVEIRA DA COSTA**, matrícula **201322010285**, apresentou, como parte dos requisitos para obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas, seu trabalho de conclusão de curso, tendo como título **"RELATO DE EXPERIÊNCIA NO MERCADO DE TRABALHO: NITRYX CONSULTING "**. Constituíram a banca examinadora os professores **Diogo Dantas Moreira** (orientador), **Ricardo de Sousa Job** (examinador) e **Fábio Abrantes Diniz** (examinador).

Após a apresentação e as observações dos membros da Banca Examinadora, ficou definido que o trabalho foi considerado **APROVADO** com nota **85**, com a condição de que o (a) aluno (a) entregue, no prazo máximo de 30 dias, a versão final do trabalho com as correções sugeridas pelos membros da banca examinadora. Eu, FÁBIO ABRANTES DINIZ, Coordenador do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, lavrei a presente ata, que segue assinada digitalmente por mim e pelos membros da banca examinadora.

Cajazeiras, 3 de fevereiro de 2022.

Documento assinado eletronicamente por:

- **Michelle Oliveira da Costa, ALUNO (201322010285) DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - CAJAZEIRAS**, em 17/02/2022 12:43:53.
- **Fabio Abrantes Diniz, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 03/02/2022 13:49:44.
- **Ricardo de Sousa Job, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 03/02/2022 13:43:09.
- **Diogo Dantas Moreira, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 03/02/2022 13:32:28.

Este documento foi emitido pelo SUAP em 18/01/2022. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 255027
Código de Autenticação: f27d2602e6

RESUMO

Este trabalho tem por objetivo apresentar as experiências adquiridas como desenvolvedor de software na empresa Nitryx Consulting durante o período de maio de 2020 a outubro de 2021 no desenvolvimento de sistemas de otimização de processos e apoio a tomada de decisão. Serão descritas informações gerais sobre a empresa, funções desempenhadas e decisões efetuadas neste período nos projetos Nitro MP e Nitro Yard. Também serão explicados o funcionamento do Processo de Desenvolvimento utilizado e as ferramentas de apoio a este processo. Por fim, serão expostas tecnologias utilizadas e em quais projetos foram aplicadas, dificuldades e aprendizados durante o período trabalhado.

Palavras-chave: Experiência. Desenvolvimento. Nitryx Consultng.

ABSTRACT

This work present the experiences acquired as a Software Developer at Nitryx Consulting during the period from May 2020 to October 2021 in the development of process optimization systems and support decision-making. There will be certain general information about the company, functions performed and decisions made during this period in the Nitro MP and Nitro Yard projects. The functioning of the Development Process used and the tools to support this process will also be explained. Finally, technologies used and in which projects they were applied, difficulties and lessons learned during the period worked will be exposed.

Keywords: Experience. Development. Nitryx.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

GHT	Gráfico Horário x Tempo
QA	Quality Assurance
FAT	Teste de aceitação na fábrica do fornecedor conforme especificação do projeto
SAT	Teste de aceitação no local onde o equipamento ou sistema foi instalado que após aprovação do cliente é lançada a versão de produção
JVM	Java Virtual Machine
GSP	Groovy Server Pages
HTTP	Hypertext Transfer Protocol

SUMÁRIO

1 INTRODUÇÃO

A Revolução Digital alterou os paradigmas do mundo, de forma que este em um curto espaço de tempo tornou-se cada vez mais conectado. Desde a execução de cálculos matemáticos em computadores gigantescos até a criação de dispositivos incrivelmente pequenos como *smartphones* e *wearables*, a humanidade vem buscando cada vez mais facilitar tarefas diárias, sejam elas dos mais variáveis âmbitos sejam transações de moeda por meios digitais, atendimento médico virtual, entre outros setores (MATTOS, 2013).

A maioria das facilidades existentes no nosso dia a dia – checar um e-mail referente a um trabalho, realizar pesquisas de forma quase que instantânea para um problema vivenciado, ocorre graças aos profissionais de tecnologia.

Dentre os profissionais de tecnologia e suas mais diversas especialidades, destaca-se para o contexto do trabalho o Desenvolvedor de Software. O desenvolvedor é o profissional responsável por solucionar problemas de mais variáveis setores, que geralmente resultam em um produto de software (U.S. NEWS & WORLD REPORT L.P, 2022).

Foi diante deste cenário que a Nitryx Consulting[1], empresa brasileira criada em 2011 a partir de um trabalho acadêmico de uns estudantes da região metropolitana de São Paulo, tornou-se uma referência por solucionar problemas relacionados a otimização de processos no setor de *supply chain*.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

[1] Site oficial da empresa: https://nitryx.com/

O objetivo geral deste documento é descrever as atividades realizadas pelo autor durante o período trabalhado na empresa Nitryx Consulting.

1.1.3 Objetivos Específicos

De acordo com o objetivo geral, são listados os seguintes objetivos específicos:

- Relatar sobre os projetos e principais atividades realizadas na empresa;
- Apresentar o processo de desenvolvimento utilizado durante o período trabalhado;
- Apresentar dificuldades encontradas durante o desenvolvimento das atribuições;

1.2 ORGANIZAÇÃO DO TRABALHO

O trabalho encontra-se organizado em 7 capítulos, de forma a atingir os objetivos descritos. No capítulo 2, são apresentados os dados gerais da empresa; no capítulo 3 serão descritas as atividades realizadas durante o período trabalhado na empresa (projetos, decisões e problemas encontrados); o capítulo 4 irá tratar de expor o processo de desenvolvimento de software utilizado na Nitryx e suas especificidades; o capítulo 5 descreverá as tecnologias utilizadas durante a execução das atividades; o capítulo 6 irá relatar as dificuldades encontradas durante o trabalho e, por fim, no capítulo 7 serão discorridas as considerações finais deste documento.

2 IDENTIFICAÇÃO DA EMPRESA

A Nitryx Consulting é uma empresa que está no mercado desde 2011 e atua no mercado desenvolvendo software por encomenda e consultoria. Sua principal área de atuação é a indústria de transportes e *supply chain*, desenvolvendo soluções de apoio à tomada de decisão - DSS. O quadro abaixo apresenta os dados da empresa:

Quadro 1 – Informações gerais sobre a empresa

CNPJ	13.351.661/0001-03
Nome da empresa	NITRYX PARTICIPACOES S.A.
Nome Fantasia	Nitryx Consulting
Endereço	Rua Barão de Jaguara, 1481 - Sala 146 - Centro, Campinas - SP, 13015-910
Telefone	+55 (19) 3217-9654
E-mail	contato@nitryx.com
LinkedIn	https://www.linkedin.com/company/nitryx-consulting

Fonte: elaborado pelo autor.

Atualmente um dos seus principais clientes é Rumo[2], mas já esteve presente no setor náutico prestando serviços a Braskem[3]. O negócio possui formato B2B. Uma empresa que trabalha com B2B – *Business-to-Business* – normalmente tem seus produtos focados em outras empresas ao invés de um consumidor individual. É comum que este formato de transação ocorra entre empresas do tipo *supply chain* e automotiva (INVESTOPEDIA, 2020).

Os produtos da empresa em sua maioria são focados no sistema ferroviário, como o *Nitro MP*, *Nitro Yard*, entre outros. Esses produtos têm como objetivo auxiliar na tomada de decisão nos mais diversos setores da ferrovia, dentre eles um dos mais importantes é a otimização da circulação de trens na malha ferroviária bem como

[2] Página oficial da Rumo: https://rumolog.com/
[3] Página oficial da Braskem: https://www.braskem.com.br/

dentro dos pátios. Mais recentemente a Nitryx foi adquirida pela Progress Rail, empresa do grupo Caterpillar (CAT)[4], devido a sua experiência em desenvolvimento de sistemas para o setor ferroviário.

A empresa possui cerca de 40 funcionários (incluindo todos os setores da empresa), especificamente na área de tecnologia 34, possuindo 3 projetos ativos: Nitro MP, Comboios e Pátio DSS, todos focados em *supply chain*. O local de trabalho de todos os colaboradores é decidido pelo próprio colaborador (uma vez que é *home office*).

Meu primeiro contato com a empresa ocorreu por meio de uma vaga divulgada no Github [5](até então não tinha o nome da empresa) informando que necessitavam de desenvolvedores Java e que fosse enviado o currículo para desenvolvedorjava2020@gmail.com. Após o envio do currículo me mandaram um teste para desenvolver utilizando Grails no *backend* e Angular no *frontend*, um sistema para armazenamento de preço das ações, aplicando alguns conceitos de desenvolvimento web como a criação de uma API, utilização de um interceptador, criação de página e de teste automatizado no *frontend* utilizando Cypress, entre outros.

[4] Aquisição da Nitryx pela Progress Rail, uma empresa do Grupo Caterpillar: https://www.progressrail.com/en/Company/News/PressReleases/ProgressRailAcquiresNitryxBrazilian Softwar AndConsultingCompany.html

[5] Página oficial do github: https://github.com/

3 DESCRIÇÃO DAS ATIVIDADES REALIZADAS

Ao entrar na empresa foi apresentado o formato de trabalho, o *home office,* e as peculiaridades que a empresa possui relacionadas a este formato de trabalho. Este modelo funciona com o auxílio de algumas ferramentas para comunicação com a equipe, sendo estas o Zoom[6] e o Slack[7], majoritariamente.

As tecnologias citadas fazem da rotina diária, uma vez que é definido uma quantidade de horas mínimas no Zoom, sendo estas 6 das 8h trabalhadas. Apesar dessa exigência, a empresa considera o horário como flexível. O Slack atua como complementar, uma forma de trocar mensagens mais duradouras e assíncronas, uma vez que mantem um histórico para consulta. Cada projeto possui sua sala no Zoom, onde os colaboradores entram em sua respectiva sala. Caso haja alguma necessidade de comunicação, a pessoa ou time será encontrado rapidamente para sanar as dúvidas. Outra questão com relação ao Zoom é que essas horas em que você esteja conectado nele faça uso da câmera pois a gerência afirma que assim a comunicação é mais efetiva, assemelhando-se ao presencial.

3.1 NITRO MP

Segmentos de forma uma ferrovia são pedaços de uma determinada ferrovia, elas são divididas nesses pedaços de forma que melhore o gerenciamento dos veículos que passam por ela, vários desses segmentos formam uma zona de controle que é comandada por um controlador, isso no Centro de Controle Operacional - CCO. Essa visualização serve para os controladores terem ciência de onde cada trem está em determinado momento e para planejar os próximos movimentos.

Em ferrovia, podemos entender conflitos como o risco de choque entre veículos que possam tender a passar no mesmo trecho no mesmo horário. Um trecho é nada mais que um pedaço da ferrovia, dentro de um determinado segmento,

[6] Página oficial do Zoom: https://www.zoom.us/
[7] Página oficial do Slack: https://slack.com/intl/pt-br/

normalmente marcado com o quilômetro. Os trens que passam por um determinado trecho devem possuir o que é chamado de licença. Licença é uma permissão que o CCO emite ao maquinista informando que este pode avançar com o veículo, sem esse gerenciamento acidentes envolvendo os transportes ferroviários pode ocorrer. O trabalho dos controladores é justamente organizar a sua zona de forma que evite esses choques e os trens cheguem aos destinos da forma mais rápida e segura possível.

O primeiro projeto no qual trabalhei foi o Nitro MP, este é um sistema que otimiza a circulação de trens na malha ferroviária. Nos primeiros dias, foram apresentadas questões referentes as regras de negócio e os termos do setor ferroviário, logo em seguida passou-se a uma segunda etapa: execução de testes no TestLodge[8] para entender melhor o funcionamento do sistema.

Antes de executar o roteiro de testes, foi feito toda a criação do ambiente para o desenvolvimento, nessa etapa seguimos um roteiro documentado de quais aplicativos deveriam ser instalados e quais customizações de configuração deveriam ser realizadas. Para este projeto o ambiente para a execução da aplicação cliente foi: Java 13 ou 11 (Azul Systems com FX), uma IDE (Eclipse ou Intellij) e para a aplicação servidora foi utilizado Java 11 (necessariamente esta versão pois as versões do Grails são bastantes acopladas a versão do Java utilizado), RabbitMQ e MySQL.

Após a execução dos testes descritos no roteiro e o ambiente devidamente configurado, deu-se início ao desenvolvimento de fato. Quando comecei no projeto as *Sprints* para a entrega do software ainda não haviam começado, então as atividades foram referentes a melhorias e mudança ao que já existia no escopo do software. O Nitro MP é um dos primeiros projetos da empresa, utilizando no módulo cliente a tecnologia Java FX, que não está mais presente no pacote *standard* do Java a partir da versão 11.

As melhorias que foram realizadas iam desde *upgrade* das tecnologias presentes no projeto à refatorações do código; além de adiantamento das tarefas mais simples que estavam presentes no pacote de entrega.

[8] Página oficial do TestLodge: https://www.testlodge.com/

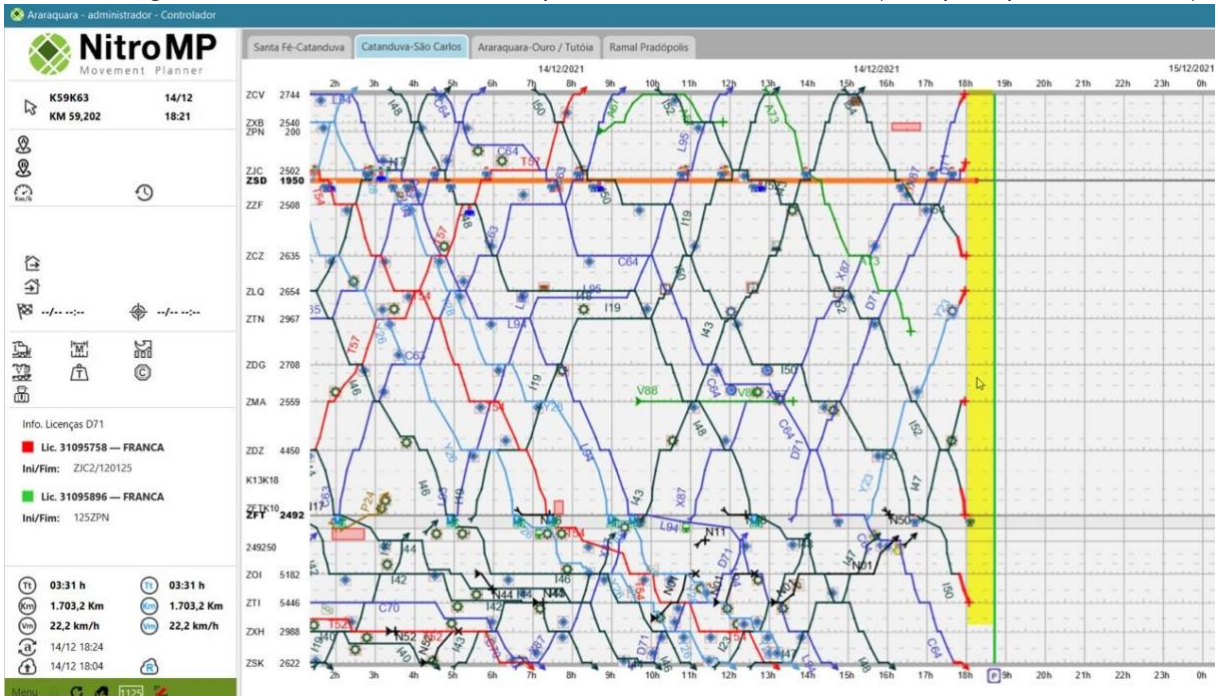
A tela principal deste software é composta por um gráfico onde em seu eixo X estão dispostas horas de um determinado dia (a configuração padrão é 1h) e além desta também é mostrado a sua data anterior e posterior. Esta configuração pode ser modificada para abranger uma maior quantidade de tempo. No eixo Y do gráfico estão dispostas os chamados Segmentos e Estações.

Cada trem pode possuir diversas atividades ao longo do percurso, estes são representados por vários ícones com diversas imagens, uma para cada tipo de atividade realizada pelo trem. Essas atividades podem ser abastecimento do veículo, troca de maquinista, entre outras. O local que o ícone se encontra indica em qual segmento a atividade será realizada. Sobre as atividades também pode ser configurado o tempo de duração.

Os retângulos destacados na tela representam as restrições de velocidade e podem apresentar 3 cores distintas: verde – para quando uma restrição não afeta grandemente a velocidade que o trem passa naquela área; amarelo – representa que o trem terá que andar com uma velocidade bastante limitada e, por fim, vermelho – representando que o trem não poderá ultrapassar aquela área enquanto a restrição estiver presente. Esses valores referentes a velocidade podem ser configurados por meio do servidor do Nitro MP, podendo conter valores padrão e valor customizado para uma restrição em específico.

Além de fornecer essa parte de visualização também gera movimentos para o planejamento do CCO, e outras funcionalidades como resolver conflitos que possam aparecer na malha ferroviária.

Figura 1 – Gráfico de trens em tempo real na malha ferroviária (Tela principal do Nitro MP).



Fonte: Elaborada pelo autor (2021).

Antes das *sprints* que relacionadas à entrega estarem definidas, começou o desenvolvimento dos itens base que foram pedidos ao cliente. Tratava-se inicialmente de inclusões de informações que eles necessitam no CCO, porém à medida que as *sprints* avançavam para a primeira entrega foram pedidas integrações com os sistemas utilizados pelo cliente, são estes o Translogic e ATW. Atuei em tarefas pontuais de desenvolvimento de telas e solução de *bugs*, porém ao iniciar as integrações eu fiquei responsável por desenvolver a maior parte delas, criando o módulo integrador.

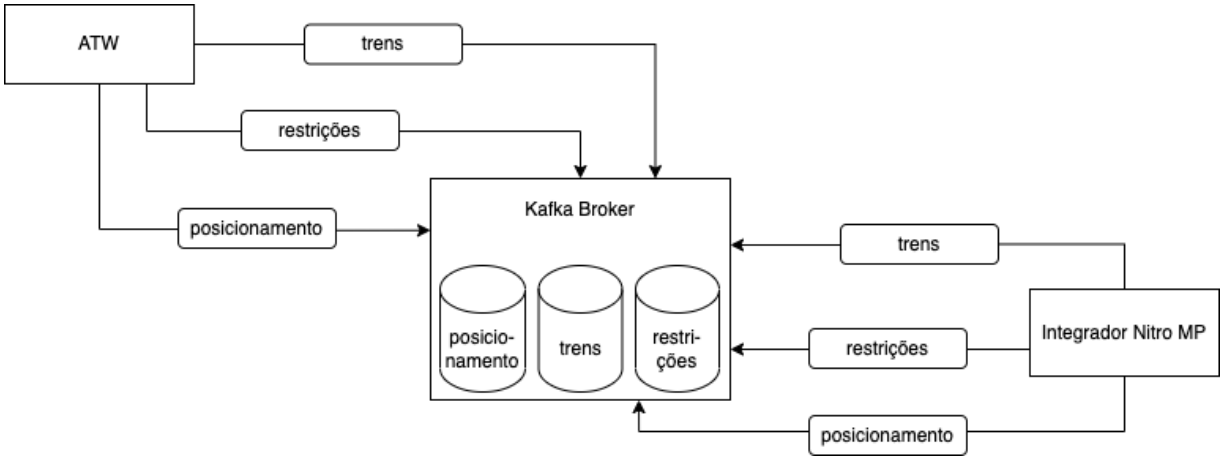
O módulo integrador inicialmente foi criado em Grails[9] pelo arquiteto, uma vez que na empresa já era uma prática comum, porém foram encontrados vários problemas de utilização do Grails com o Apache Kafka, pois o *plugin* grails-micronaut-kafka possuía funcionalidades limitadas para trabalhar com essa tecnologia. Após várias tentativas falhas de utilizar Grails na integração, houve uma conversa com os arquitetos para utilizarmos Java e Spring Boot, pois todos os nossos problemas com relação a conversão dos objetos proveniente dos tópicos Kafka não aconteceriam,

[9] Página oficial do Grails Framework: https://grails.org/

além de permitir uma maior manipulação de *Acknowledge* e eventos que poderiam nos ajudar a detectar problemas de indisponibilidade, entre outros. Assim, o módulo integrador foi reescrito.

As mensagens eram enviadas pelas aplicações externas a empresa por meio de vários tópicos Kafka, de forma a serem consumidas pelo integrador do Nitro MP. Os dados eram guardados em tópicos de forma que o integrador pudesse-as consumir sem a necessidade de uma comunicação direta com os demais sistemas. O projeto de integração possuía um consumidor para cada tópico. A figura 2 mostra a interação entre a Producer API e Consumer API de um tópico Kafka presente em um Kafka Broker, onde o produtor é o sistema ATW e o Integrador Nitro MP o consumidor das mensagens.

Figura 2 – Produção e consumo de mensagens por meio tópicos Kafka.



Fonte: Elaborada pelo autor (2022).

O módulo servidor do projeto – responsável por armazenar os dados referentes aos trens e passar os dados para o módulo cliente – também utilizava Grails (que continua sendo utilizado até hoje), porém como o sistema é antigo todas as páginas deste era feito em GSP e estava previsto no sprint a modernização dessas páginas, foi feita uma migração destas páginas e *framework* escolhido pelos arquitetos foi o Angular. As páginas GSP foram sendo trocadas pelas reescritas utilizando a nova

tecnologia enquanto as antigas que não foram migradas continuavam sendo utilizadas. Até o momento que eu trabalhei no projeto ainda não haviam sido migradas todas as páginas do sistema.

Nesse aspecto, além de desenvolver as páginas novas trabalhei criando *tasks* ao build.gradle, arquivo que configura a automatização de tarefas no Grails Framework, pois o Grails utiliza o Gradle para todo o gerenciamento de *build*. Assim, ao executar a aplicação servidora as páginas estariam inseridas neste pacote, sem a necessidade de executar o angular em outro host / porta. A configuração necessária para este *build* encontra-se na figura abaixo:

Figura 3 – Configuração do angular dentro do projeto Grails.

```
node {
    version = '12.18.3'
    download = true
    nodeModulesDir = file("src/main/e-graph-web")
}

clean {
    file('src/main/webapp/web').deleteDir()
    file('src/main/e-graph-web/.node_modules').deleteDir()
}

task serveClient(type: NpmTask, dependsOn: 'npmInstall') {
    group = 'application'
    description = 'Run client using ng serve'
    args = ['run', 'start']
}

task buildClient(type: NpmTask, dependsOn: 'npmInstall') {
    group = 'build'
    description = 'Compile client side assets for production'
    args = ['run', 'build']
}

task buildClientWatch(type: NpmTask, dependsOn: 'npmInstall') {
    group = 'application'
    description = 'Builds and watches the client side assets for rebuilding'
    args = ['run', 'buildWatch']
}

task clientTest(type: NpmTask, dependsOn: 'npmInstall') {
    group = 'verification'
    description = 'Executes client side unit tests'
    args = ['run', 'test']
}

task clientIntegrationTest(type: NpmTask, dependsOn: 'npmInstall') {
    group = 'verification'
    description = 'Executes client side integration tests'
    args = ['run', 'e2e']
}

bootRun.dependsOn(buildClientDev)

bootWar.dependsOn(buildClient)
```

Fonte: Elaborada pelo autor (2021).

Na entrega do primeiro pacote (chamada de versão 5.1), percebeu-se pela equipe de QA problemas referentes ao serviço de mensagem utilizado na comunicação cliente-servidor, no caso o RabbitMQ. No ecossistema do Nitro MP o RabibtMQ[10] é utilizado para envio de mensagens do módulo servidor (que recebe

[10] Página inicial do RabbitMQ: https://www.rabbitmq.com/

mensagens do módulo integrador) para o módulo cliente. Apenas o módulo integrador utiliza o Kafka pois os dados da integração são entregues por este meio. Assim, foram feitas algumas alterações de forma que esses dados não fossem perdidos. Para a descrição dos bugs apresentados entende-se mensagem como todo e qualquer dado enviado por meio de uma *Queue* no RabbitMQ.

Alguns dos bugs relatados foram: ao suspender o sistema com o cliente aberto por um período de 1h aproximadamente o RabbitMQ não estabelecia mais a conexão, parando o envio e recebimento de mensagens. Para solucionar este problema foi criado um mecanismo de tolerância à falhas para identificar a desconexão do RabbitMQ e reconectá-lo. Outro bug corrigido ocorria ao criar uma mensagem a ser enviada por um cliente e um outro cliente estivesse com falta de internet, ao cliente reestabelecer a conexão ele não receberia essa mensagem pendente, apesar das filas que este está conectado ainda estivessem “vivas”. A solução apresentada pelo time foi salvar o nome das filas que foram criadas na comunicação, pois, apesar das filas estarem ativas para o determinado *host*, a aplicação cliente não conseguia mais acessá-las.

Outro problema corrigido, ocorria ao estar com o cliente conectado após 6 horas seguidas as filas deixavam de funcionar, mesmo que estas estivessem identificadas pelo cliente e a conexão com o *broker* estabelecida. Isso acontecia devido ao tempo de vida dessas filas em si, a solução foi alteração em alguns parâmetros dessas filas como o TTL – Time To Live, que realiza a remoção da fila fosse feita quando ocorre um determinado período de inatividade, além de aumentar o tempo de duração da mensagem (parâmetro expires).

3.2 NITRO YARD

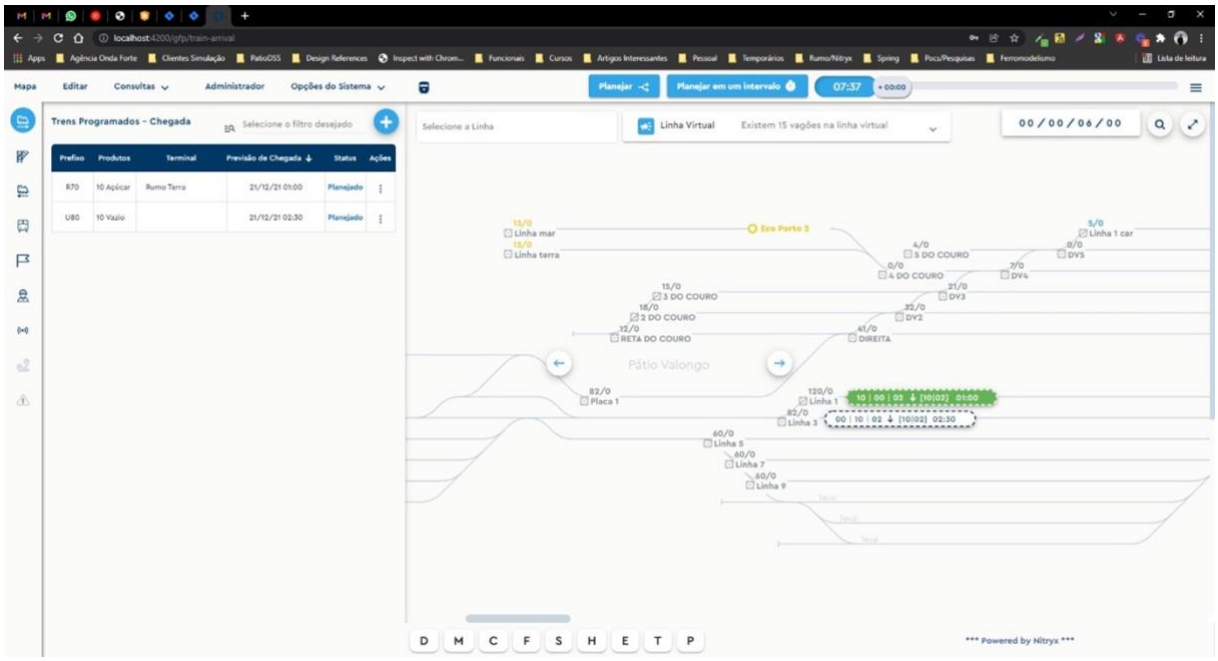
Nitro Yard ou Pátio DSS, este é focado nos Pátios Ferroviários, isto é, no processo de carga e descarga dos vagões dentro destes. Esse mostra informações de quais terminais cada trem entrou e toda movimentação com os veículos (vagões e locomotivas), da sua entrada a sua saída do Pátio. Todo o processo em relação ao

entendimento das regras de negócio e configuração do ambiente de desenvolvimento é semelhante em todos os projetos.

Neste projeto a minha contribuição foi relacionada em sua maioria a correção de *bugs* na aplicação. Diferentemente do Nitro MP, este é um projeto relativamente novo que acabou de ter seu primeiro pacote entregue, as tecnologias utilizadas neste são Oracle Database, Groovy com Spring Boot no servidor, o integrador foi desenvolvido com Java com Spring Boot e Apache Kafka para comunicação com os sistemas externos.

O Nitro Yard tem a aplicação cliente feito em Angular, a comunicação cliente-servidor ocorre por meio de requisições HTTP (Hypertext Transfer Protocol) e Web Sockets. Os gráficos deste incluem um de terminais de descarga e um no formato GHT (disposição semelhante ao Nitro MP, onde tem os terminais ferroviários contrapostos à data/horário), onde delimita os terminais e os locais que uma composição (conjunto de vagões e locomotivas) foi movido ao longo do tempo. Também possui um gráfico chamado de GFP (mostra a movimentação de vagões e locomotivas dentro do terminal). A figura 2 mostra a tela inicial do GFP.

Figura 4 – Gráfico de movimentação de composições no terminal.



Fonte: elaborada pelo autor (2021).

O processo de desenvolvimento utilizado é o mesmo para todos os projetos da empresa, trabalhei em sua maioria corrigindo bugs de interface e no servidor, porém com outra equipe. De funcionalidades desenvolvidas neste projeto foi feito a criação de *login* por meio do OpenID[11].

[11] Página oficial do OpenID: https://openid.net/

4 PROCESSO DE DESENVOLVIMENTO

A Engenharia de Software tem por finalidade solucionar problemas relacionados diversas áreas de estudo e, geralmente, estes resultam em um produto de software.

Segundo (PRESSMAN; MAXIM, 2016, p. 26)

> Todo projeto de software é motivado por alguma necessidade de negócios – a necessidade de corrigir um defeito em uma aplicação existente; a necessidade de adaptar um “sistema legado” a um ambiente de negócios em constante transformação; a necessidade de ampliar as funções e os recursos de uma aplicação existente ou a necessidade de criar um novo produto, serviço ou sistema.

Para a criação de um produto de software, faz-se necessário que exista um processo definido, processo este que possui certa complexidade, uma vez que o produto obtido precisa atender de forma viável as demandas no qual este foi proposto a suprir.

Segundo Sommerville (2011, p.5) “Um processo de software é uma sequência de atividades que leva à produção de um produto de software.” Assim, o processo de desenvolvimento de software é composto por inúmeras etapas que possuem extrema importância para o êxito na solução do problema proposto, bem como altos custos relacionados à falhas em cada fase de desenvolvimento.

No âmbito de Processos de desenvolvimento de Software, destaca-se uma categorização, esta divide-se em processo planejado e processo ágil. Processos planejados são aqueles onde todos os passos são arquitetados anteriormente a fase de desenvolvimento, tendo sua eficácia medida de acordo com este planejamento inicial. Os processos ágeis têm sua organização de forma diferente, uma vez que estes são mais flexíveis devido ao seu planejamento gradual, tornando mais simplificado adequar o projeto as mudanças que possam vir a ocorrer.

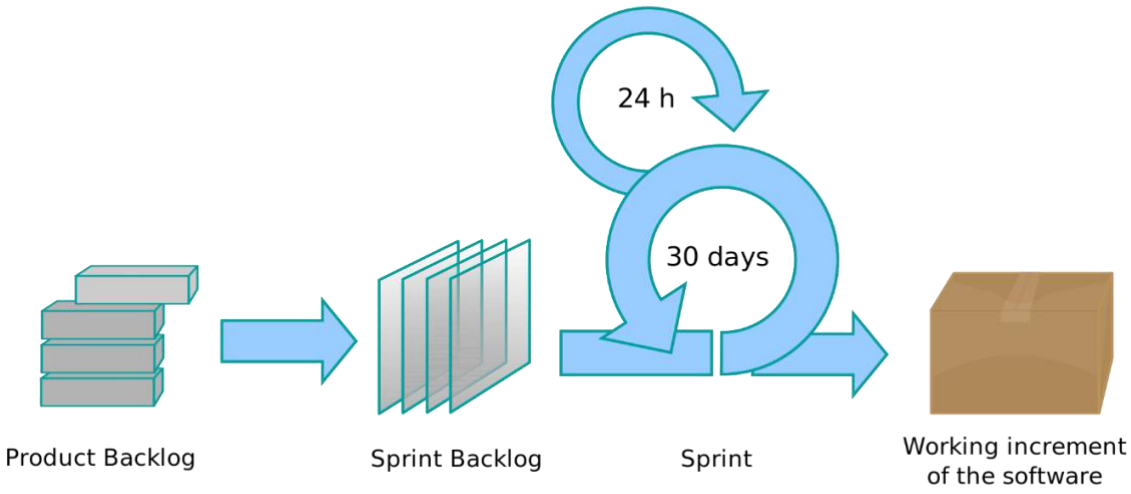
O Movimento Ágil[12] tem um papel importante na formação dos processos de desenvolvimento atuais, uma vez que este tornou possível essa maior flexibilização às mudanças no escopo de um projeto de software, pois os processos planejados tornavam os custos de mudanças e evolução muitas vezes inalcançáveis.

Na empresa são adotados o Scrum como metodologia de desenvolvimento e o Jira Software como ferramenta para apoio ao gerenciamento de projeto.

4.1 SCRUM

O Scrum pode ser definido como um método ágil de desenvolvimento de software. Este método é caracterizado pela sua iteratividade, onde cada ciclo de desenvolvimento neste método é chamado de *Sprint*. A figura 2 mostra de forma geral como ocorre o funcionamento do Scrum.

Figura 5 – Processo de Desenvolvimento com Scrum.



Fonte: artigo sobre a metodologia Scrum no site Treasy[13].

[12] Manifesto para Desenvolvimento Ágil de Software: https://agilemanifesto.org/iso/ptbr/manifesto.html
[13] Disponível em: <https://www.treasy.com.br/blog/scrum/>. Acesso em: 28 nov. 2021.

A fase inicial de um projeto Scrum é a definição do *Product Backlog*, onde serão descritos os trabalhos que devem ser realizados na execução do projeto, tarefas que normalmente serão selecionadas para desenvolvimento em uma *Sprint*. A decisão de quais tarefas serão executadas em um determinado ciclo é dada por meio dos chamados *stakeholders*. Os *stakeholders* são as pessoas interessadas no projeto: clientes, pessoas relacionadas ao custo e ao desenvolvimento.

O *Sprint* é a unidade principal do desenvolvimento utilizando Scrum, possuindo um comprimento fixo de acordo com cada projeto. No início de cada um desses ciclos são discutidos qual trabalho deverá ser feito, quais as etapas de avaliação do mesmo e os recursos necessários para que as etapas sejam efetuadas.

No Scrum existe o conceito de *roles*. As *roles* ou papeis definem as responsabilidades dentro de um projeto gerenciado através do Scrum, são estes *Product Owner*, *Scrum Master* e *Developer Team*.

O *Scrum Master* tem por responsabilidade assegurar que as práticas do Scrum sejam seguidas, além de atuar como facilitador da comunicação entre os membros da equipe.

Já o *Product Owner* tem como uma de suas funções de priorizar quais trabalhos tem maior prioridade no ciclo de desenvolvimento, atuando como um elo entre o cliente e o *Developer Team*. Deve possuir um profundo conhecimento sobre as regras de negócio, pois deve saber esclarecer quaisquer detalhes para o time de desenvolvimento e os clientes.

Por fim, o *Developer Team* possui o papel de transformar as tarefas designadas para a Sprint em um Produto de Software, não sendo formado apenas por desenvolvedores, mas também pelo time de *Quality Assurance* (QA) e o de infraestrutura.

Outro conceito considerado um pilar para o Scrum são os rituais. Estes são o *Sprint Planning*, *Daily Stand-up*, *Iteration Review* e *Restrospective*. Estes podem ser visto com mais detalhes no quadro 2.

Quadro 2 – Rituais Scrum e seus objetivos

Ritual	Objetivo	Frequência de Realização
Sprint Planning	Selecionar as tarefas do *Product Backlog* que serão desenvolvidas na *Sprint*, bem como o esforço necessário para realizá-las.	Início da *Sprint*
Daily Stand-up	Informar a todos o andamento das atividades da Sprint, de forma a identificar impedimentos e prevenir atraso das entregas.	Uma vez ao dia
Iteration Review	Demonstração do trabalho efetuado na *Sprint* e obter feedback dos interessados no Projeto.	Final da *Sprint*
Retrospective	*Feedback* sobre o desenvolvimento da *Sprint*, tornando possível o entendimento sobre o que está ou não trazendo benefícios para o time.	Final da *Sprint*

Fonte: elaborada pelo autor (2021).

De acordo com cada empresa e cada projeto estes rituais podem variar, serem incluídos alguns outros ou removido, de acordo com as necessidades do projeto e da equipe.

Na Nitryx Consulting, o *Iteration Review* normalmente não é adotado pois existe um tempo de Sprint Reservado para FAT (testes realizados pela equipe de QA no ambiente de desenvolvimento) e SAT (testes realizados pela equipe de QA no ambiente de homologação), onde o time de QA juntamente com *Product Owner* faz a verificação das tarefas desenvolvidas na Sprint, de forma a garantir que os requisitos foram atendidos e que não existem erros nas funcionalidades desenvolvidas.

4.2 JIRA SOFTWARE

Para apoiar o Processo de Desenvolvimento e o Gerenciamento de Projetos existem inúmeras ferramentas no mercado. Uma delas é o Jira Software[14], ferramenta da Atlassian, criado em 2002 inicialmente era um software para rastreamento de bugs.

Esta ferramenta tem aplicabilidade em vários formatos de gerenciamento de times com metodologias ágeis como Scrum e Kanban. A proposta da ferramenta é centralizar toda essa parte de monitoramento de tarefas em um único local, facilitando o rastreamento de bugs, gerenciamento de tarefas com a possibilidade de utilizar fluxos de trabalho customizáveis, entre outras funcionalidades.

As colunas no *board* do Jira utilizadas no dia a dia dos projetos da empresa no Scrum Board normalmente eram: Tarefas Pendentes, Em Desenvolvimento, *Rework*, *To Merge*, *To Test, Testing* e *Done.* Cada uma dessas colunas definem uma fase do desenvolvimento na qual as tarefas estavam no momento.

Ao sair do Sprint Backlog, a primeira coluna que uma tarefa passa é para a Tarefas Pendentes, isto indica que esta tarefa ainda normalmente ainda não foi atribuída a um desenvolvedor ou inicializou-se ainda o desenvolvimento.

Uma tarefa que foi iniciada por um desenvolvedor passa para a coluna de Em Desenvolvimento. Nessa fase todas as implementações na aplicação e configuração necessária são realizadas. Uma tarefa que sai dessa coluna normalmente passa para *To Merge.*

Na coluna *To Merge* todas as tarefas que estão em suas *branchs* definidas pelo seu número do Jira, como por exemplo [MP-180], são atualizadas para a *branch* de desenvolvimento (que na empresa normalmente utiliza-se o número da versão para tal), passando a ficar em espera dos testes da equipe de QA.

Normalmente no board de desenvolvimento existem 3 colunas utilizadas pela equipe de QA, são elas: *Rework*, *To Test* e *Testing*. A coluna *To Test* é definida por

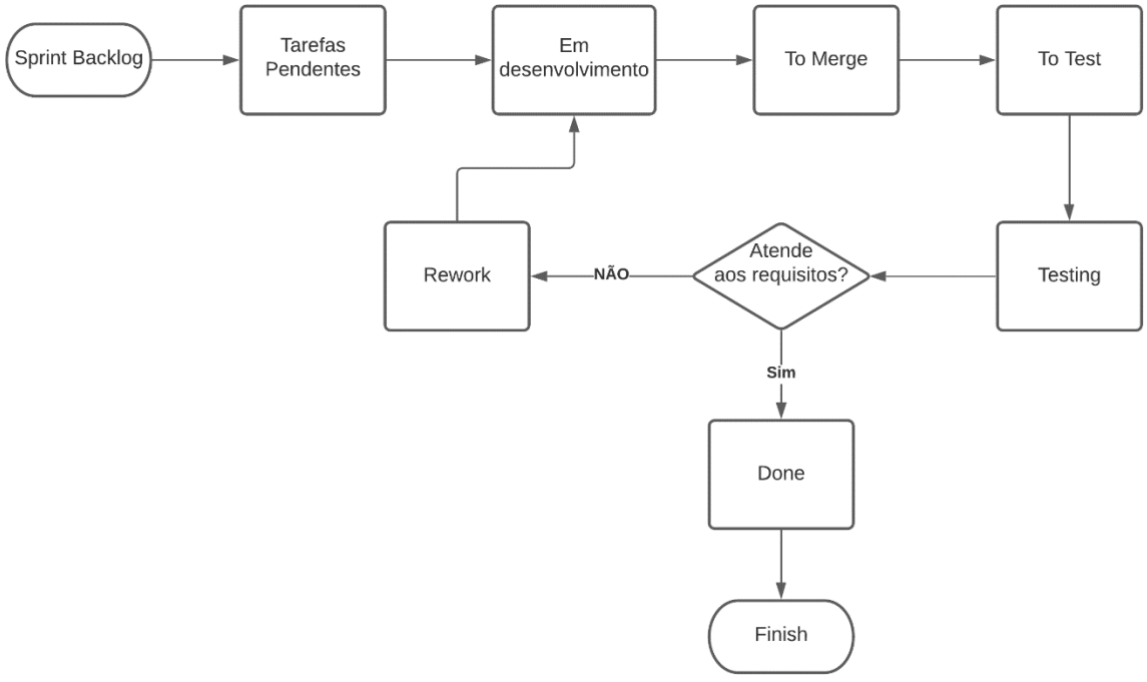
[14] Jira Project Management Software. Disponível em <https://www.atlassian.com/software/jira/agile>.

tarefas que ainda não tiveram as funcionalidades / correções testadas pela equipe de QA. Ao pegar uma tarefa para realizar os testes ela passa a coluna *Testing*. Uma tarefa em "*testing*" pode ter 2 destinos. Se a tarefa estiver nos conformes dos requisitos ela passa para *Done*, caso contrário ela vai para *Rework*.

Tarefas em *Rework* precisam ser reimplementadas para atender aos requisitos do sistema, voltando para "Em desenvolvimento" enquanto o desenvolvedor realiza os ajustes na mesma, voltando para todo o ciclo descrito anteriormente.

Enfim, tarefas que tiveram seus requisitos atendidos na fase de testes passam para a coluna *Done*, informando que estão prontas para fechamento da versão nas quais elas estão empacotadas. A figura 3 descreve o fluxo de trabalho de acordo com as colunas descritas:

Figura 6 – Fluxo de trabalho de acordo com o Scrum Board da Nitryx Consulting.



Fonte: Elaborada pelo autor (2021).

5 TECNOLOGIAS UTILIZADAS

As tecnologias são uma parte importante no Desenvolvimento de Software pois é com elas que são implementadas as soluções de *software*. Assim, a escolha de ferramentas e tecnologias para o desenvolvimento pode impactar diretamente no êxito da solução apresentada com relação a escalabilidade e outros fatores no desenvolvimento. Esta seção apresenta as tecnologias que foram utilizadas no dia a dia na empresa.

5.1 GRAILS FRAMEWORK

O Grails é um framework para construção de aplicações web com sua primeira versão liberada em 2006. O foco desse framework seria uma alta produtividade pois utiliza-se de "codificação por convenção", assim como é visto no Ruby on Rails[15], porém utilizando Groovy, uma linguagem feita para executar utilizando a JVM (Java Virtual Machine).

Esse framework utiliza-se de ferramentas do Java como o Hibernate, Spring, Java Server Pages, entre outros recursos da linguagem Java com uma pré-configuração das ferramentas, a fim de que o desenvolvedor possa focar mais na implementação da solução. Atualmente o Grails utiliza como base do framework o ecossistema Spring (Portal GSTI, [S. l.], 2021).

O Grails é bastante utilizado no ambiente da empresa desde a versão 2.0 nas aplicações Back-End. Sistemas como o Nitro MP que estão na empresa desde o começo utilizaram não no desenvolvimento das funcionalidades do sistema, mas também toda a parte da camada de apresentação com o GSP[16], que funciona da mesma maneira como o JSP com renderização *server-side*, porém utilizando a linguagem Groovy. Nos projetos mais novos e nas novas versões que são

[15] Site oficial do Ruby on Rails: https://rubyonrails.org/
[16] Introdução ao Groovy Server Pages: https://gsp.grails.org/latest/guide/index.html

desenvolvidas no Nitro MP, as páginas web não são mais implementadas no lado do servidor, tendo sido substituído pelo framework Angular, que será tratado posteriormente neste capítulo.

Atualmente os projetos utilizam a versão 4 do Grails Framework, tendo como obrigatoriedade a utilização do Java Development Kit 11, pois além das versões do framework serem atreladas a uma versão do Java este utiliza-se do Spring-Loaded[17] para fazer alterações em tempo de execução no código executado como a alteração de variáveis, construtores e outros elementos presentes no código da aplicação Grails.

5.2 SPRING BOOT

A IBM Cloud Education (2020) afirma que o Spring Boot é uma ferramenta que torna o desenvolvimento de aplicativos da web e mais rápido e simplificado por utilizar uma autoconfiguração, permitindo ao desenvolvedor não precisar realizar a adição de dependências, exceto quando há a necessidade de uma customização, além de aplicações Spring executar sem a necessidade de um Servidor ou Container de Aplicação.

As aplicações de integração entre sistemas no Nitro Yard e Nitro MP foram desenvolvidas utilizando o ecossistema Spring – tecnologias e frameworks criados para implementar recursos de especificações Java de forma a prover uma integração sem a necessidade de sempre reimplementar comportamentos padrão.

Além dos integradores, o Nitro Yard também utiliza o Spring Boot, com o diferencial deste utilizar o framework com linguagem Groovy. O Nitro MP utiliza na aplicação servidora de forma indireta uma vez que o Grails 4 utiliza o Spring Boot na sua implementação.

[17] Página do Github do projeto Spring Loaded: https://github.com/spring-projects/spring-loaded

5.3 JAVA DEVELOPMENT KIT

O Java Development Kit ou simplesmente JDK é um pacote de ferramentas utilizadas para o desenvolvimento de aplicações na linguagem Java, provendo aos desenvolvedores a implementação das especificações da plataforma Java, incluindo o compilador da linguagem e as bibliotecas que fazem parte das especificações. Além disso, ela traz consigo o Java Virtual Machine (JVM) que provê o gerenciamento de memória e o Java Runtime Enviroment (JRE), responsável por fazer a execução dos programas Java. (InfoWorld, [S. I.], 2021).

Essa tecnologia possui um destaque especial porque as aplicações Back-End de todos os projetos utilizam como base o JDK, uma vez que mesmo as aplicações que não escritas em Java – como o Groovy – e as que utilizam o framework Grails, são executadas por meio da JVM.

Além disso, o projeto Nitro MP possui um módulo desenvolvido com a especificação Java Standard Edition (Java SE). Este módulo constitui a aplicação cliente que se comunica com o servidor através de requisições HTTP e pelo serviço de mensagem RabbitMQ, que será descrito posteriormente. A versão utilizada do JDK pela aplicação atualmente é a 13, a implementação utilizada da JDK é provida pela Azul Systems – empresa que implementa a versão *open source* da JDK, (Open JDK), já que a interface é construída com o Java FX[18], e a Oracle deixou de incluir a especificação a partir JDK 11.

5.4 ANGULAR

Segundo Google (2010) o Angular é uma plataforma para desenvolvimento de aplicações *single-page* escrita utilizando Typescript[19]. Ela provê uma estrutura baseada em componentes por meio de uso de várias bibliotecas que compõem a

[18] Página oficial do Java FX: https://openjfx.io/
[19] Página oficial do Typescript: https://www.typescriptlang.org/

plataforma, com ferramentas para vários estágios do desenvolvimento como configuração, teste, desenvolvimento e outros.

A plataforma possui uma versatilidade muito grande, uma vez que seu código pois além de ser utilizado em aplicações web é facilmente adaptado para ambientes *desktop* e *mobile*. Devido a este contexto que o angular foi escolhido pelos arquitetos dos projetos na empresa.

As atividades desenvolvidas utilizando angular dentro do Nitro MP foi em sua maioria a recriação das páginas outrora existentes no servidor Grails – utilizando GSP (Groovy Server Pages) – para a aplicação angular, de forma a modernizar a aplicação tanto em questão de tecnologia pois agora as páginas não mais seriam renderizadas no lado do servidor bem como em questão de usabilidade – modernização do UX/UI da aplicação, que normalmente era realizado pelos desenvolvedores Front-End. Todas as novas telas que foram desenvolvidas para a aplicação eram criadas no projeto angular, projetadas de forma que caso haja necessidade, seja feito o redirecionamento para as páginas GSP. No projeto do Nitro Yard não cheguei a trabalhar no desenvolvimento de nenhuma página da aplicação focando apenas na correção de bugs da aplicação.

6 DIFICULDADES ENCONTRADAS

Ao iniciar na empresa uma das dificuldades mais destacáveis foi entender o funcionamento do modelo de trabalho utilizado, o home office. Não necessariamente o fato de trabalhar de casa, mas sim acostumar-se com a obrigatoriedade de estar conectado utilizando a câmera do computador por meio da reunião no Zoom ao menos 6 das 8h diárias. Porém, à medida do transcorrer do tempo acabei acostumando-me com essa exigência da empresa.

Destaca-se, também, como uma dificuldade, apesar desta ser menos considerável, foi entender certos fluxos específicos do Grails e as incompatibilidades apresentadas de uma versão para outra, mesmo quando a mudança se caracterizava como *minor*[20], como da versão 4.0 para 4.1, tornando o update do framework no projeto um pouco oneroso, apesar de que o teste de admissão na empresa foi implementação de um cenário utilizando esta tecnologia.

Por fim, pode-se enumerar também a falta de documentação em algumas partes do sistema, como o a tela principal do cliente do Nitro MP, tornando qualquer mudança muito custosa quando não pudesse ter acesso ao responsável pelo desenvolvimento desta.

Essa dificuldade dar-se principalmente pelo fato que toda a construção da interface com Java FX foi feita manualmente, com variáveis, valores e expressões matemáticas sem uma semântica evidente e sem comentários no código, tornando certas partes do código quase se incompreensíveis, por exemplo uma variável DY = 5.6, uma variável sem um nome expressivo tornando difícil a compreensão de sua função dentro do escopo.

[20] Versionamento semântico – O que é e como usar: https://imasters.com.br/codigo/versionamento-semantico-o-que-e-e-como-usar

7 CONCLUSÃO

A cada nova experiência, sempre extraímos grandes conhecimentos, com o mercado de trabalho não poderia ser diferente. Os aprendizados foram em muitos aspectos, tanto com relação ao funcionamento de uma ferrovia – algo que eu jamais teria a chance de entender se não fossem os projetos trabalhados – assim como entender na prática os papeis dentro do desenvolvimento, pois anteriormente não havia obtido experiência de uma empresa grande que possui papeis bem separados e definidos, desde pessoas que trabalham apenas em qualidade de software bem como como um *product owner*.

Vivenciar um ambiente com um processo de desenvolvimento bem definido auxilia muito a desempenhar o seu papel como desenvolvedor, já que as tarefas presentes no *board* já continham caso de uso e entradas e saídas esperadas. Os *bugs* normalmente já vinham com os passos executados para a reprodução do erro, pois existia um time de qualidade para realizar esses testes. Essas pequenas divisões de tarefas que para um leigo pode parecer apenas mais recursos alocados numa empresa, fazem total diferença na confiança dos produtos e serviços prestados.

Com relação a obtenção de informações relacionadas a este trabalho, a empresa foi bastante solicita ao me responder dúvidas referentes ao projeto, principalmente pelo fato que no momento eu ainda fazia parte do time da empresa. Pode-se concluir também que poderia ter uma reformulação no modelo de trabalho, uma vez que a empresa já tem bastante maturidade neste formato, acredito que poderia ter mais liberdade para o desenvolvedor pois considero negativa a câmera ser mandatória.

Por fim, foi muito importante para a formação como profissional, pois trabalhei com tecnologias que outrora não tive oportunidade de conhecer.

REFERÊNCIAS

ATLASSIAN. **What is Jira used for?** Disponível em: <https://www.atlassian.com/software/jira/guides/use-cases/what-is-jira-used-for#Jira-for-requirements-&-test-case-management>. Acesso em: 7 dez. 2021.

GOOGLE. **Angular, 2010.** The modern webdeveloper's platform. Disponível em: <https://angular.io>. Acesso em: 28 nov. 2021.

IBM CLOUD EDUCATION. **Java Spring Boot.** 2020. Disponível em <https://www.ibm.com/cloud/learn/java-spring-boot>. Acesso em: 7 dez. 2021.

INFOWORLD. **What is the JDK? Introduction to the Java Development Kit.** Disponível em: <https://www.infoworld.com/article/3296360/what-is-the-jdk-introduction-to-the-java-development-kit.html>. Acesso em: 7 dez. 2021.

INFOWORLD. **Built for realtime: Big data messaging with Apache Kafka, Part 2**. 2018. Disponível em: < https://www.infoworld.com/article/3066873/big-data-messaging-with-kafka-part-2.html>. Acesso em: 12 dez. 2021.

INVESTOPEDIA. **Business-to-Business (B2B).** 2020. Disponível em: < https://www.investopedia.com/terms/b/btob.asp>. Acesso em: 14 dez. 2021.

MATTOS, Sérgio Augusto Soares. **A revolução digital e os desafios da comunicação.** Cruz das Almas: Editora UFRB, 2013.

ORACLE. **JDK 11 Release Notes**. Disponível em: <https://www.oracle.com/java/technologies/javase/11-relnote-issues.html>. Acesso em: 28 nov. 2021.

PORTAL GSTI. **Grails**. Disponível em: <https://www.portalgsti.com.br/grails/sobre/>. Acesso em: 7 dez. 2021.

PRESSMAN, R. S.; MAXIM, B. S. **Engenharia de Software: uma abordagem profissional.** 8. ed. Porto Alegre: AMGH, 2016

REVISTA /DEV/ALL. **Grails: 10 anos depois – Um Pouco de História – Primeira Parte.** Disponível em <https://revista.devall.com.br/grails-10-anos-depois-um-pouco-de-historia-primeira-parte/>. Acesso em 7 dez, 2021.

U.S. NEWS & WORLD REPORT L.P. **Software Developer.** Disponível em: < https://money.usnews.com/careers/best-jobs/software-developer>. Acesso em: 10 fev. 2022.

SOMMERVILLE, I. **Engenharia de Software**. 9. ed. São Paulo: Pearson, 2011.

WILKER, Jeremy. **Angular in Action.** New York: Manning Publications, 2018.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
Campus Cajazeiras
Rua José Antônio da Silva, 300, Jardim Oásis, CEP 58.900-000, Cajazeiras (PB)
CNPJ: 10.783.898/0005-07 - Telefone: (83) 3532-4100

Documento Digitalizado Ostensivo (Público)

Documento Final TCC

Assunto: Documento Final TCC
Assinado por: Michelle Oliveira
Tipo do Documento: Dissertação
Situação: Finalizado
Nível de Acesso: Ostensivo (Público)
Tipo do Conferência: Cópia Simples

Documento assinado eletronicamente por:

- **Michelle Oliveira da Costa**, **ALUNO (201322010285) DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - CAJAZEIRAS**, em 14/03/2022 09:50:32.

Este documento foi armazenado no SUAP em 14/03/2022. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 460323
Código de Autenticação: 95542c9c19