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

DE GRADUANDO A ANALISTA: DESENVOLVIMENTOS E PERSPECTIVAS SENDO ANALISTA

JULIERME JADON OLIVEIRA MANGUEIRA

Cajazeiras
2021

JULIERME JADON OLIVEIRA MANGUEIRA

DE GRADUANDO A ANALISTA: DESENVOLVIMENTOS E PERSPECTIVAS SENDO ANALISTA

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. Ricardo de Sousa Job.

Cajazeiras
2021

FICHA CATALOGRÁFICA

Dados Internacionais de Catalogação na Publicação (CIP)

M277g Mangueira, Julierme Jadon Oliveira

De graduando a analista: desenvolvimentos e perspectivas sendo analista/Julierme Jadon Oliveira Mangueira. – Cajazeiras/PB: IFPB, 2021.

72f.: 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. Ricardo de Sousa Job.

1. Desenvolvimento de sistemas 2. Análise de sistemas 3. Billing and Revenue Management (BRM) 4. Tecnologia da informação

I. Mangueira, Julierme Jadon Oliveira II. Titulo

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 16h00 do dia 20 do mês de DEZEMBRO do ano de 2021, o(a) aluno(a) **JULIERME JADON OLIVEIRA MANGUEIRA**, matrícula **201412010233**, 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 **"DE GRADUANDO A ANALISTA: DESENVOLVIMENTOS E PERSPECTIVAS"**. Constituíram a banca examinadora os professores **Ricardo de Sousa Job** (orientador), **Diogo Dantas Moreira** (examinador) e **George Candeia de Sousa Medeiros** (examinador).

Após a apresentação e as observações dos membros da Banca Examinadora, ficou definido que o trabalho foi considerado **APROVADO** com nota **100**, 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, 24 de janeiro de 2022.

Documento assinado eletronicamente por:

- **Julierme Jadon Oliveira Mangueira**, **ALUNO (201412010233) DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - CAJAZEIRAS**, em 18/02/2022 08:48:04.
- **Diogo Dantas Moreira**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 03/02/2022 13:32:12.
- **George Candeia de Sousa Medeiros**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 31/01/2022 09:29:31.
- **Ricardo de Sousa Job**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 24/01/2022 08:13:08.

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: 255020
Código de Autenticação: 1e8bc567ea

AGRADECIMENTOS

Primeiramente a Deus, por ter me dado saúde, coragem e força de vontade na superação dos obstáculos durante essa caminhada e a intercessão que recebi para que eu tivesse a saúde mental em dia para escrever e finalizar esse documento.

Aos meus pais, José Aldevam e Jeanne, que me incentivam a continuar crescendo, amadurecendo e buscando concluir mais uma etapa em minha vida. Minha família teve mais papel importante no meu crescimento e, algumas cobranças para que eu finalizasse esse trabalho, meu sincero agradecimento a cada um de coração. Se eu cheguei até aqui, só tenho a agradecer a família que eu tenho. Representadas aqui por Francisca Cabral de Oliveira *(In Memorian)* e Francisca Saraiva Mangueira, *(In Memorian)*.

Obrigado a cada um dos meus amigos que fizeram parte da minha formação e continuam presentes em minha vida, vocês que sempre me incentivaram a alcançar voos maiores e mais distantes.

Meu agradecimento a toda equipe da Teccel Engenharia e Teccel Energia Solar por me acolher como estagiário e, por ter feito parte dessa história. A todos os meus incríveis colegas da Accenture, representados na pessoa de Francisco Sousa, e ao meu time, Squad Faturamento II, vocês me ensinam bastante e, sou muito grato a cada um Vini, Ana, João, Heráclito, Pamella, Igor, Raphael, Rubem e Michel. Não sei onde o futuro me levará, todavia saibam que estando aqui me sinto como parte de uma família.

Ao Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, campus Cajazeiras por me proporcionar um ambiente criativo, acolhedor e de bastante aprendizado.

Ao meu atual orientador Ms Prof. Ricardo de Sousa Job, por ser paciente em todo o processo de elaboração deste trabalho, pelo seu incentivo, conselhos e sua dedicação para comigo. Aos meus orientadores de projetos passados, Prof. Danilo Lucena e Ms. Prof. George Candeia, se vocês não tivessem acreditado nas minhas ideias malucas, eu não teria chegado aqui. Meu nobre antigo professor e amigo, Genemes, que me fez renovar a fé para buscar novas oportunidades de trabalho.

E, por último, uma homenagem para todo o corpo docente e colegas do curso de Análise e Desenvolvimento de Sistemas:

Fábio meu professor de programação que me ensinou o Pascalzão
Ele me disse uma vez
"Julierme, tente outra vez"
Eu já tava bem saturado de toda aquela coisa
Tava faltando só a corda pra eu poder fugir
Ademar meu professor de computação
que me ensinou os binarão
Ele me disse certa vez
"Julierme, fique de boa porque tudo é lindo"
E assim vou seguindo
Marcando falta com o Pascalzão
Fábio ficou chateado porque
eu parei de programar o Pascalzão
Eu disse para ele então
"Fábio, irmão, não aguento mais esse Pascal não"
Vagner tava passando
"Julierme, não chateie Fábio não"
Virginia que tava no canto falou então
"Fábio, deixa o rapaz que ele tá
sobrecarregado com tanta confusão"
E eu então respondi
"Mas, que confusão? Fábio é que gosta do Pascalzão"
Anrafel estava no corredor indo para sua aula
Quando ouviu o debate:
"Não se afobe não, que as respostas estão na Integração"
Janderson ainda tentou questionar que eram
Nos Banco de Dados que se guardavam todas as respostas
Mas Barros quem bateu o martelo e disse por fim:
"Na Orientação de Objetos que está o nosso presente, Julierme"

Em um outro dia,
Humberto chegou apresentando para turma as topologias das redes
João Paulo demonstrou como a sociedade e a TI se comportam
Mas foi Diogo quem disse por fim:

"Aplicações só se comportam assim
Quando construímos um excelente projeto de sistemas"
"Mas para se construir precisamos antes
Conhecer algumas das Estruturas de Dados, meus alunos", disse André.
Gustavo sabido como era
"Exatamente, meu colega de trabalho,
Não vou deixar esse barco afundar ou não saberemos como mexer nas árvores"
Genemes não deixou barato
"Não se esqueçam que estamos na Web e aqui precisamos ser perspicazes"
Flávia chegou concordando
"Já precisamos colocar essas perspectivas em um artigo"
Eva que já estava na sala, acrescentou:
"Vamos colocar essas mentes para pensar além da caixinha"
"Essa super caixa que ela está se referindo é seu Sistema Operacional"
Chegou Atanasio palestrando juntamente com George
"Não precisamos ir longe,
pensem que essa perspectiva roda numa camada criada pelo seu SO"
Ari chegou cheio de moral e explicou:
"Isso também se aplica ao mundo móvel, turma,
Mas cuidado com a perspectiva adotada nesse universo nano"
Paulo logo em seguida nos alertou:
"Os banco de dados não convencionais se acoplam perfeitamente ao seu projeto
Basta escolher um, estudar e aplicar"
Job ouvia a tudo aquilo como quem ouve música e argumentou:
"Em um universo corporativo o cuidado tem que ser redobrado!
Por que não levar estes ensinamentos para a vida?"

Crescer dentro do IFPB - Cajazeiras foi transformador
Aprendi com os professores, a direção e toda a equipe
E cada um deles foi bastante importante para mim
E acredito que para meus colegas também
Esta é uma pequena homenagem a cada um...
Que guardarei seus ensinamentos em meu SO numa pastinha especial
Por fim, então vem ADS o curso que se tornou parte de minh’alma
E que será muito difícil nos separar.

”Abençoado pela chuva que cai
Você consegue imaginar onde a vida pode te levar?
Agora que você está rezando pra todos
Você está tão mal?

Fique, não vá, você precisa fazer uma escolha”

Andre Matos

RESUMO

O destemido trabalho apresenta a experiência enquanto graduando e profissional na área de Tecnologia da Informação, mais especificamente como Analista Júnior. Até conseguir esse emprego, precisou trabalhar em três empresas na cidade de Cajazeiras, em diferentes setores, desde Estagiário, a Webmaster e gerente de redes sociais até chegar a ser Freelancer e, por fim, alcançar o tão esperado cargo de Analista em uma empresa de Campina Grande que tem se expandido principalmente durante a pandemia. Cada uma delas trouxera novas perspectivas para sua carreira. E, para a escrita deste documento, a empresa escolhida foi a Accenture Technology. Visto que, aqui estarão descritos sobre algumas das experiências com as tecnologias utilizadas juntamente com suas descrições baseado no que foi desenvolvido, tomando como base o projeto que possibilitou adquirir novos conhecimentos e, trilhar por novos horizontes, este foi o Billing and Revenue Management (BRM). Durante essa caminhada, muitas dificuldades foram vencidas, trazendo consigo um significativo aprendizado.

Palavras-chave: Analista. Desenvolvimento de Software. Metodologias Ágeis. Projeto. BRM.

ABSTRACT

The fearless work presents the experience as an undergraduate and professional in the Information Technology field, more specifically as a Junior Analyst. Until he got this job, he had to work in three companies in the city of Cajazeiras, in different sectors, from Intern, Webmaster and Social Media Manager to become a Freelancer and, finally, reach the long-awaited position of Analyst in a company of Campina Grande that has expanded mainly during the pandemic. Each had brought new perspectives to his career. And, for the writing of this document, the chosen company was Accenture Technology. Since, here we will describe some of the experiences with the technologies used together with their descriptions based on what was developed, based on the project that made it possible to acquire new knowledge and tread new horizons, this was Billing and Revenue Management (BRM). During this journey, many difficulties were overcome, bringing with them significant learning.

Keywords: Analyst. Software development. Agile Methodologies. Project. BRM.

LISTA DE FIGURAS

LISTA DE CÓDIGOS

LISTA DE ABREVIATURAS E SIGLAS

OMS	Organização Mundial da Saúde
XP	Extreme Programming
PO	Product Owner
TSC	Time de Scrum
TD	Time de Desenvolvimento
SM	ScrumMaster
BRM	Billing and Revenue Management
ECE	Elastic Charging Engine
PDC	Pricing Design Center
CM	Connection Manager
SL	Shared Libs
DM	Data Manager
TFVC	Controle de versão do Team Foundation
US	Histórias de Usuário
TU	Testes Unitários
TS	Testes de Histórias
DACC	Débito Automático em Conta Corrente
TXT	Arquivo de Texto

SUMÁRIO

1 INTRODUÇÃO

Durante a construção da carreira profissional na área de Tecnologia da Informação, uma carreira em constante evolução, a oportunidade de trabalhar nos mais diversos segmentos: empresa de energia solar, *marketing, freelancer* e, por último, na área de tecnologia. Nesse processo, ocorreram oportunidades para se conhecer e aprender com inúmeras pessoas dos mais variados níveis de instrução e formação.

No decorrer desse período de atuação no mercado profissional, foi difícil para se conseguir uma oportunidade, a fim de que pudesse aplicar todos os conceitos e práticas aprendidos durante a graduação. Visto que, na de energia solar, o trabalho de estagiário se concentrou no desenvolvimento do *website*, auxílio na parte de informática, *Marketing Digital* e, por último, o gerenciamento do processo de construção de uma aplicação móvel, já na empresa de *marketing*, o trabalho consistiu no gerenciamento de redes sociais, cujo cargo era de *Webmaster*; como *freelancer* houve diversos tipos de demanda, sendo os mais recorrentes, o de criação de arte digital e o desenvolvimento de *websites*. O foco deste trabalho não consistirá em descrever tais experiências, por não terem tido tanto foco no quesito desenvolvimento de software.

Atualmente, por estar vinculado regularmente a uma empresa de tecnologia, conseguiu-se adquirir e lapidar de forma mais intensa os conhecimentos desenvolvidos durante a graduação, para apresentar toda essa vivência profissional, cuja finalidade deste documento é demonstrar parcialmente essa experiência adquirida junto a um grande projeto na empresa Accenture, onde o discente exerce a função de Analista de Sistemas Júnior.

Como caso de uso, o projeto apresentado aqui será a tecnologia de Billing and Revenue Management (BRM), que corresponde a um sistema completo de cobrança, faturamento e gerenciamento de receitas. Nas sessões seguintes serão apresentados os conceitos, tecnologias e processos utilizados no desenvolvimento desse projeto, bem como algumas histórias mapeadas e codificadas.

O trabalho tem como objetivo apresentar as atividades desenvolvidas em um ambiente profissional, colocando em prática todo o conhecimento fomentado durante o período da graduação, mesclando a teoria com as práticas, para prover soluções mais adequadas à empresa. Ao passo que é uma oportunidade ideal para se absorver mais conhecimentos e experiências em um ambiente altamente produtivo.

A empresa escolhida foi a Accenture do Brasil, a que fica localizada em Campina Grande, cujo foco é o desenvolvimento de soluções corporativas. O trabalho ágil e crescente desenvolvido possibilitou que algumas dessas atividades fossem demonstradas. O time que serviu de base para a realização das atividades foi o de Ciclo da Receita e Faturamento II.

Este documento apresenta a seguinte organização: no segundo capítulo temos uma explanação maior sobre a Accenture, o terceiro apresenta os processos de desenvolvimento de software e metologias ágeis, no quarto detalhes da tecnologia de BRM e principais softwares utilizados, o quinto serão demonstradas algumas das atividades realizadas e, por último, as considerações finais do trabalho.

2 A ACCENTURE

A Accenture é uma empresa multinacional com capital aberto de consultoria de gestão, tecnologia da informação e no setor industrial. É a maior empresa de consultoria do mundo, além de ser uma competidora global no setor de inovação tecnológica. Fornece serviços de estratégia e consultoria (ramo *Strategy & Consulting*), tecnologia (ramo *Technology*), operações (ramo *Operations*) e, por último, o ramo *Interactive*, que entrega: *marketing* digital, *marketing* analítico e gestão de *mídia* (ACCENTURE, 2021).

Figura 1 – Logo da Accenture

Fonte: Site da Accenture

Desde 2006, a Accenture está na lista do Guia Você S/A-Exame como uma das 150 melhores empresas para trabalhar no Brasil, a melhor em desenvolvimento de carreira e, classificada em segundo lugar entre as Vinte Companhias Internacionais mais Sustentáveis, da Barron. E, atualmente no *ranking* em trigésima quarta colocação com uma das Empresas Mais Admiradas do Mundo, na Fortune.

A empresa foi fundada em 1989 em Chicago, EUA (como *Andersen Consulting*), atualmente ela possui uma sede incorporada em Dublin, na Irlanda e, tem uma visão bem delineada, nas palavras de Julie Sweet (Presidente global):

> Mundo afora, uma coisa é universalmente verdadeira a todas as pessoas da Accenture: nós nos preocupamos com o que fazemos e com o efeito do que entregamos aos nossos clientes e comunidades. É questão pessoal para todos nós (ACCENTURE, 2021)

A Accenture se posiciona valorizando os valores essenciais por meio dos comportamentos individuais, afinal são eles os pilares para toda forma de agir e tomar decisões. Ainda agir com ética e integridade, cuja cultura na tomada de decisão contra o racismo, a inclusão de pessoas com deficiência, além de possuir e apoiar projetos sociais e ser grande aliada da comunidade LGBTQIA+. Só no Brasil, a Accenture tem um quadro de mais de 15.000 empregados, com escritórios em diversas partes do

país: São Paulo, Rio de Janeiro, Recife, Brasília, Belo Horizonte, Nova Lima, Vitória, Campina Grande, Porto Alegre e São Bernardo do Campo.

O destaque fica para o Nordeste, em especial Recife, com um quadro de mais de 3 mil funcionários e que possui ainda o *Innovation Center* no Porto Digital. O setor *Accenture Technology* é o que mais tem crescido exponencialmente; desde 2017, vem apostando em serviços de Agile, DevOps, Nuvem e Mobilidade; firmando parcerias e trazendo novos projetos de inovação.

Foi justamente devido a esse clima pandêmico que o setor tecnológico, em todo o mundo, evoluiu de forma espantosa, a procura por profissionais no setor triplicou praticamente, as empresas que antes não dispunham de um setor de Tecnologia da Informação (TI) precisou se reestruturar e pequenos negócios também aderiram ao mundo digital (COSTA, 2021).

E, devido à pandemia do SARS-CoV-2, causador da doença Covid-19, todas as relações presenciais precisaram ser restritivas, adotando prioritariamente o trabalho remoto, como uma das medidas preventivas seguindo as orientações da OMS[1].

No que tange à Paraíba, a companhia possui um escritório em Campina Grande há 12 anos, cidade considerada como um dos polos tecnológicos, a segunda maior cidade do estado com mais de 400 mil habitantes e que todos os anos realiza o maior São João do mundo. Acompanhando esse fluxo, a Accenture Campina Grande precisou aumentar deliberadamente seu quadro de funcioáários para atender a toda a demanda, passando de 200 funcionários em 2019, para mais de 300, sendo 30% mulheres.

[1] Organização Mundial da Saúde.

3 PROCESSOS DE DESENVOLVIMENTO

Um processo de *Software* é um conjunto de operações relacionadas cujo resultado é a produção de um sistema (software); elas podem ser feitas do zero fazendo uso de uma linguagem de programação (C, Java, C#, entre outras linguagens). No entanto, *softwares* corporativos não seguem essa regra, geralmente, são desenvolvidos através de extensões ou apenas modificando sistemas já existentes; como, por exemplo, adicionar um assistente virtual a um sistema legado[2] (SOMMERVILLE, 2019).

De acordo com Sommerville (2011), existem os mais diversos tipos de processos de *software*, porém eles compartilham de quatro pilares fundamentais:

- Especificação
- Projeto e implementação
- Validação
- Evolução

Na teoria se resume a estas quatro; todavia, na prática o cenário muda, visto que dentro de cada pilar existem ainda subatividades específicas como mapeamento dos requisitos, arquitetura do *software*, testes unitários etc. E, há ainda atividades que são complementares ao processo, como a documentação e o gerenciamento de configuração.

Existem os mais diversos tipos de processos de desenvolvimento de *software*: métodos tradicionais [cascata] e os métodos ágeis [*Extreme Programming* (XP), *Scrum, Crystal Clear, Feature Driven Development* e outros]. Neste trabalho, trataremos apenas dos métodos ágeis, citando apenas o XP e o *Scrum*.

Nas décadas de 80-90 (SOMMERVILLE, 2019), existia uma visão generalizada de que a melhor maneira para conseguir atingir a excelência em *software* era garantida por meio de um planejamento minucioso do projeto, segurança formalizada, uso de métodos de análise e projeto assegurado por ferramentas CASE (*Computer-aided software engineering*) e, é claro, todo o processo de desenvolvimento era rigoroso e controlado. Essa visão gerou bastante insatisfação por ser uma abordagem engessada

[2] Sistemas legados são *softwares* obsoletos que atualmente não existe mais suporte a eles ou, por alguma razão, se tornou inviável a sua manutenção.

e que acabava por comprometer todo o desenvolvimento quando surgia alguma nova mudança.

A própria comunidade sugeriu, em meados da década de 90, métodos que permitissem que a equipe focasse no *software* em si, e não dedicado em sua concepção e documentação. Sommerville ainda deixa delimitado que métodos ágeis são baseados em uma abordagem puramente incremental para a especificação, o desenvolvimento e a entrega do *software*.

Sommerville (2019, p.60) ainda complementa que:

> Os métodos ágeis são mais adequados para desenvolver aplicações nas quais os requisitos do sistema mudam rapidamente durante o processo. Eles se destinam a fornecer rapidamente um software funcional para o cliente, que, por sua vez, pode propor a inclusão de requisitos novos ou modificados nas iterações seguintes. Eles visam reduzir a burocracia do processo ao evitar o trabalho com valor duvidoso no longo prazo e eliminar a documentação que provavelmente jamais será utilizada.

Existem ainda alguns princípios ágeis que norteiam o processo de desenvolvimento de modo a responder mais às mudanças do que ficar preso a um plano de execução (AUDY, 2015). São estes:

1. A prioridade é a satisfação do cliente, através de entregas contínuas de *software*;
2. Estar por dentro das mudanças de requisitos, mesmo estando em fase final do desenvolvimento. Podendo acontecer de surgirem demandas mais urgentes.
3. Entregar *software* funcionando com frequência (*release*), sendo a escala adotada de forma quinzenal.
4. Um grupo de pessoas relacionadas aos negócios e o time de desenvolvimento trabalham em conjunto e diariamente, durante todo o curso do projeto.
5. Construção de projetos ao redor de indivíduos motivados, trazendo para o time o ambiente e o suporte necessário.
6. O método mais eficiente e eficaz de transmitir informações para, e por dentro de um time de desenvolvimento, é feito através de reuniões remotas.
7. *Software* funcional é a medida primária de progresso.
8. Processos áis promovem um ambiente sustentável. Passos constantes devem ser mantidos indefinidamente.

9. Atenção contínua à excelência técnica e aos bons princípios de design para aumentar essa agilidade.
10. Sempre que possível manter a simplicidade para maximizar a quantidade de trabalho que não precisou ser feito.
11. Nivelar o time para ser sempre autoorganizável.
12. Em intervalos regulares, o time se reúne para refletir os pontos positivos e negativos que aconteceram e, logo em seguida, ajusta-se e se otimiza seu comportamento.

3.1 XP

Para Gomes (2014, p. 14) o método ágil *Extreme Programming*, popularmente conhecido como XP, foi criado e inicialmente divulgado por Kent Beck em meados da década de 90, considerado por muitos como o método ágil que tem melhor cobertura em aspectos técnicos como codificação, *design* e testes. Durante o processo de desenvolvimento há quatro atividades básicas a serem executadas: Ouvir, Desenhar, Codificar e Testar.

Gomes (2014) reforça que:

> É preciso ouvir porque desenvolvedores nem sempre possuem o conhecimento de negócio necessário para se construir o software. O *Planning Game* é uma reunião que acontece uma vez a cada iteração, em que o principal objetivo é decidir quais funcionalidades serão desenvolvidas na iteração e aprender mais sobre elas. O cliente escreve histórias de usuário em cartões que representam a necessidade de funcionalidade a ser desenvolvida, e explica para os desenvolvedores tudo o que for preciso para que eles possam implementá-la.

O principal objetivo é manter o *design* sempre simples, evitando sempre que possível a complexidade com a criação de funcionalidades que não sejam realmente triviais. Ao manter o *design* simples e coeso, automaticamente, faz com que lhe poupe tempo.

Não se pode fugir da codificação já que é uma atividade essencial para o desenvolvimento de *software*; afinal de contas, sem as instruções codificadas não existe *software* algum.

Testes são atividades de extrema importância para garantir a qualidade do *software* e não é algo opcional com XP, ao contrário, todo código deve possuir seu

teste de unidade. Quando um defeito é encontrado, cria-se um teste de unidade para assegurar que esse mesmo problema jamais volte a acontecer. Com o XP, os testes são elaborados antes da codificação de produção através de TDD (*Test Driven Development*, traduzindo Desenvolvimento Guiado por Testes) (GOMES, 2014).

3.2 *SCRUM*

Dentro do cenário de gestão de desenvolvimento de *software* é comumente utilizado o *Scrum* como principal método adotado. Como definição (SABBAGH, 2014), *Scrum* é um *framework* [2] ágil para a gestão do desenvolvimento de *softwares* complexos imersos em ambientes complexos onde cada iteração é chamada de *Sprint*. O *Scrum* é bastante parecido com o XP, porém este vai além dos valores e princípios ágeis, já que ele também tem o seu próprio conjunto de princípios a serem seguidos. São eles que complementam as regras do *Scrum* descritas em seus papéis, artefatos e eventos; os princípios são:

1. Foco: Evitar a multitarefa, o time trabalhando focado em metas de negócios claras e alcançáveis com as quais se comprometeram e, em apenas um projeto por vez.
2. Coragem: Abraçar as mudanças como parte natural do processo de desenvolvimento, confiança no time e deixá-lo livre para realizar o trabalho necessário para atingir metas acordadas.
3. Franqueza: O time busca validar os resultados do seu trabalho a partir do *feedback* sobre o que produzem, isso ajuda a criar visibilidade sobre as mudanças necessárias.
4. Compromisso: O time é que deve determinar como seu trabalho será realizado, monitorando seu progresso e realizando as correções de rumo quando forem necessárias.
5. Respeito: Todo os membros que compõem o time trabalham juntos, compartilhando responsabilidades e garantindo uma maior colaboraąão uns com os outros em seu trabalho.

De acordo com Sabbagh (2014, p.59), dentro do Scrum é necessário visualizar os papéis que cada pessoa desempenha no time:

[2] Framework, nesse contexto, é uma série de ações e estratégias que visam à solução de um problema específico.

Como primeiro personagem, o ***Product Owner* (PO)** é único e é o responsável por definir, comunicar e manter a Visão do Produto relativamente constante ao longo do projeto. Ele é quem está alinhado com os clientes do projeto e com quaisquer outras partes interessadas que possam colaborar para o entendimento e definição do Produto.

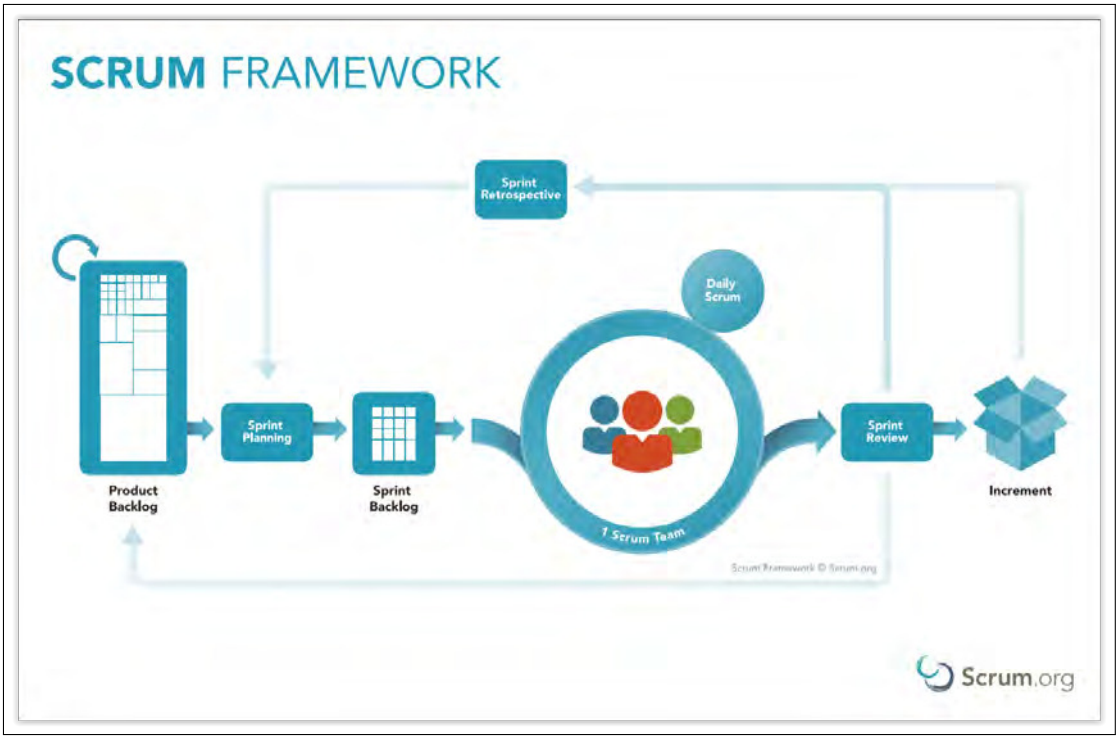
Logo abaixo vem o **Time de *Scrum* (TSC)** composto por: um Time de Desenvolvimento e um *ScrumMaster*. O **Time de Desenvolvimento (TD)** ou os desenvolvedores são eles que definem como irão realizar o trabalho e gerenciar seu progresso em direção a metas de negócios acordadas com o PO. E, por último, não menos importante, o **ScrumMaster (SM)** é responsável por facilitar que as obstruções que possam acontecer dentro do Time de Desenvolvimento e que encontrem em seu trabalho sejam removidas, facilita as interações entre o time e o PO e precisa se manter neutro.

Sobre sprint, Gomes (2014, p.13) nos apresenta uma descrição bastante assertiva, onde:

> Toda sprint começa com uma reunião de planejamento, que é dividida em duas partes. A primeira delas tem um enfoque mais estratégico, na qual se decide o que será feito, isto é, quais funcionalidades serão implementadas e define-se uma meta para a Sprint. Para isso, o PO apresenta os itens do product backlog, e a equipe obtém informações suficientes sobre cada um dos itens, dessa forma, é possível fornecer uma estimativa que expresse um tamanho ou um número de horas para o trabalho e assim seja definido quantas e quais tarefas poderão ser desenvolvidas no sprint (de acordo coma velocidade da equipe que é calculada através de dados de sprints passados).

Para a segunda parte da reunião, são definidas quais tarefas serão desenvolvidas na *sprint*, cujo enfoque é mais tático, de modo a decidir como serão feitas as tarefas, para só então se analisar tarefa por tarefa ficando atentos aos detalhamentos técnicos. Mediante isso, será possível tomar diversas decisões de negócio e decisões técnicas (GOMES, 2014). Finalizada a reunião de planejamento, o TD então começa a trabalhar nas tarefas, respeitando as prioridades, fazendo sempre primeiro as tarefas mais importantes, aquelas que foram alinhadas com o PO. Um cronograma então é organizado sobre as entregas e todos os dias é realizada uma reunião do TSC para conversar sobre o andamento da sprint, batizada de *Daily* (Figura 02).

Figura 2 – Fluxograma do Scrum



Fonte: Scrum.org, 2021

Com esse *framework* temos uma visão geral, um planejamento de *release* em que acontece todo o mapeamento do que os usuários desejam ou necessitam para gerar uma estimativa de complexidade e linha do tempo. O detalhamento técnico acontece de cada vez, o suficiente para as próximas semanas, sempre trabalhando na porção de prioridades (AUDY, 2015).

Audy (2015) completa: “para tanto, é preciso criar um *continuum* em que, sempre ao terminar uma *Sprint*, já tenhamos o detalhamento da próxima para que possa ser estimado e desenvolvido”. Ou seja, enquanto uma *Sprint* está sendo em evolução, em paralelo, já se tenha alguma definição sobre a próxima.

3.3 SCRUM APLICADO NO PROJETO

A Accenture por ser uma multinacional atua nos mais diversos tipos de projetos e com as mais diversas metodologias, desde cascata até as mais ágeis. Mesmo estando ligado a um dos escritórios da empresa, ocorre o contato com pessoas de outros escritórios, projetos, bem como dos mais diversos tipos de habilidades e nacionalidades.

O projeto que serviu de norte para esse trabalho está diretamente ligado ao Faturamento, antes disso, não houveram quaisquer atividades dentro da Accenture. Cabe ressaltar ainda que existem diversos pontos que não foram relatados neste trabalho por irem em desacordo com o ponto de confidencialidade e segurança que exige a empresa.

Assim como no SCRUM, existe nesse projeto um conjunto delimitado de *sprints* que fazem parte da *Program Implementation, ou PI*, é feito um refinamento de *Features* e *US* que serão desenvolvidas durante cada *sprint*, cada uma dessas histórias são pontuadas e selecionadas para só depois serem mapeadas e incluídas nas *sprints*. Esses refinamentos são realizados com a presença do arquiteto, PO, os SMs e o time de desenvolvimento.

Em cada sprint acontecem as *Dailys* realizadas com a presença de dois SMs (um do time e outro do projeto) que avaliam o andamento da *sprint* e, caso ocorra algum impedimento, ele seja mapeado e mitigado. Ao final da sprint, o time realiza então a validação das USs juntamente com o PO e os dois SMs, por fim, depois de aceitas que serão incluídas na release que estas foram programadas.

4 TECNOLOGIAS

Este capítulo delimitará as tecnologias e softwares que foram utilizados no decorrer do presente relato de experiência. A linguagem de programação utilizada pelo projeto é C, porém a tecnologia que será apresentada mais à frente, possui sua própria estrutura; no banco de dados utilizou-se do SQL e, para alguns processos será utilizada a estruturação/codificação Unix. Apenas a título de curiosidade durante todo este relato o sistema operacional utilizado para rodar os diferentes softwares necessários para o projeto foi o Microsoft Windows©.

4.1 BRM

BRM (Billing and Revenue Management) é uma tecnologia que agrega um sistema completo de cobrança, faturamento e gerenciamento de receitas para provedores de serviço (ORACLE, 2020). Quase todas as políticas de negócios já têm uma implementação padrão. Por exemplo, o ciclo de faturamento padrão é de um mês.

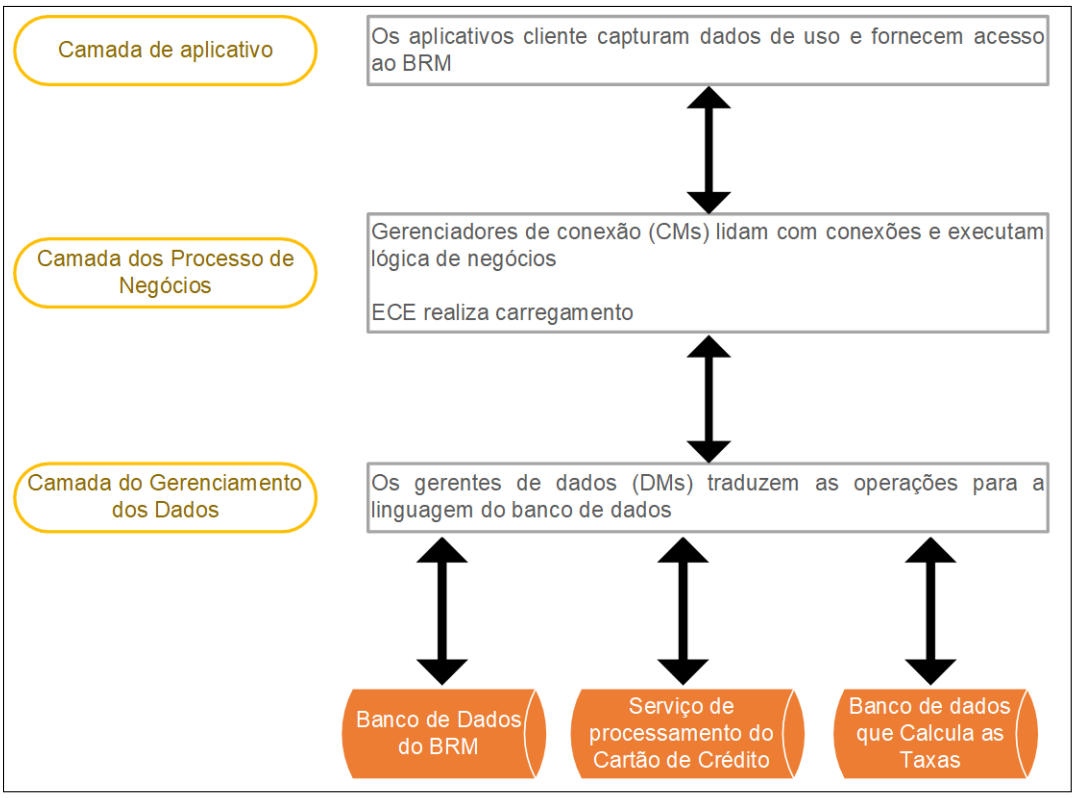
Alterações são possíveis editando arquivos de configuração ou personalizando o código da política respectiva a esse processo. Onde a maioria dos recursos do BRM pode ser personalizada sem qualquer programação.

Este sistema agrega como funções principais:

- A *Cobrança (Charging)* que determina quanto cobrar de um cliente pelo uso do serviço e taxas recorrentes.
- O *Faturamento (Billing)* compila os impactos do saldo em uma fatura, geralmente a cada mês.
- O *Processamento de Pagamentos (Payment processing)* exige um pagamento do cliente.
- O *Gerenciamento de Clientes (Customer management)* cria contas para os assinantes e gerencia seu status.

A arquitetura da próxima página, figura 3, mostra como os componentes BRM processam um nome de login quando um usuário inicia um aplicativo cliente BRM.

Figura 3 – Arquitetura do Sistema de BRM



Fonte: Elaborado pelo autor, 2021

Com a Suite do BRM é possível:

- **Cobrar dos clientes pelo uso do serviço e assinaturas.** Permite configurar cobrança online (por exemplo, para avaliar chamadas pré-pagas em tempo real) e offline, que avalia o uso registrado em arquivos CDR.

  O uso é classificado pelo *Elastic Charging Engine (ECE)* ou pelo *Pipeline Manager*. Os encargos de assinatura (por exemplo, taxas mensais recorrentes) são avaliados pelo servidor próprio do BRM no ECE. O próprio usuário pode definir cobranças, criar ofertas de produtos no *Pricing Design Center (PDC)* ou até mesmo listas de preços dentro do PDC.

- **Gerenciar saldos de clientes.** Conforme o uso é avaliado, os saldos dos clientes são atualizados em tempo real. É permitido executar uma variedade de contas e tarefas que serão demonstradas nos saldos do cliente, aqui é possível fazer ajustes e fornecer reembolsos.

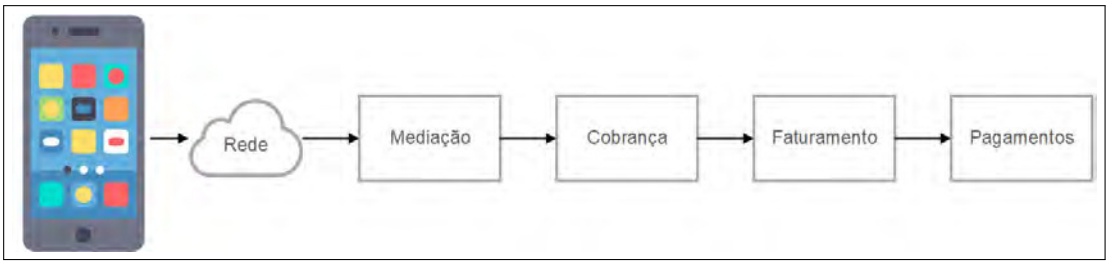
- **Crie contas e receba pagamentos.** O BRM por si só compila todos os impactos ao realizar um balanço em cima das contas dos clientes. É possível usar uma

variedade de métodos de pagamento, como cartão de crédito, fatura por e-mail, boleto para solicitar pagamentos.

- **Gerenciar contas de clientes.** As contas são armazenadas no banco de dados BRM. Estão disponíveis os métodos de criação, desativação e fechamento contas e gerenciamento das compras do cliente e ofertas de produtos.
- **Colete inteligência de negócios.** É possível utilizar os relatórios do BRM para analisar os usos de um determinado cliente e planejar suas ofertas.

Exemplificando: Um cliente abre uma chamada, esta é atendida por um sistema de mediação, que se conecta ao ECE que por sua vez realizará a classificação. Então, após essa classificação, aplicará a cobrança, o ECE irá atualizar o saldo do cliente no banco de dados BRM. E quando ocorre a execução do faturamento, o próprio BRM cria uma fatura que incluirá as faturas cobradas no saldo do cliente. Ele também cria uma solicitação de pagamento e a envia ao cliente, conforme apresentado na Figura 4.

Figura 4 – Visão geral das funções de negócios do BRM



Fonte: Elaborado pelo autor, 2021

Ao adotar o BRM como sistema, antes é necessário realizar as principais configurações que são:

- Configurar os componentes do sistema; por exemplo, conectar um *Connection Manager (CM)* a um *Data Manager (DM)* configurando um nome de host e número de porta.
- Configurar opções de lógica de negócios; especificar o dia do mês de cobrança padrão, é uma delas.
- Carregar dados usados pelo BRM; IDs de razão geral, por exemplo.

Partindo para a parte de desenvolvimento, o BRM é integrado ao *SQL Developer* como Banco de Dados e, como é utilizado uma versão em nuvem do BRM, então

como ferramenta optou-se pelo *Openshift* para monitoramento e gerenciamento desse sistema. Usa-se ainda o *Azure DevOps* para os desenvolvedores em si, uma vez que o código é integrado ao ambiente de desenvolvimento e é compilado e testado, todos serão expostos mais adiante.

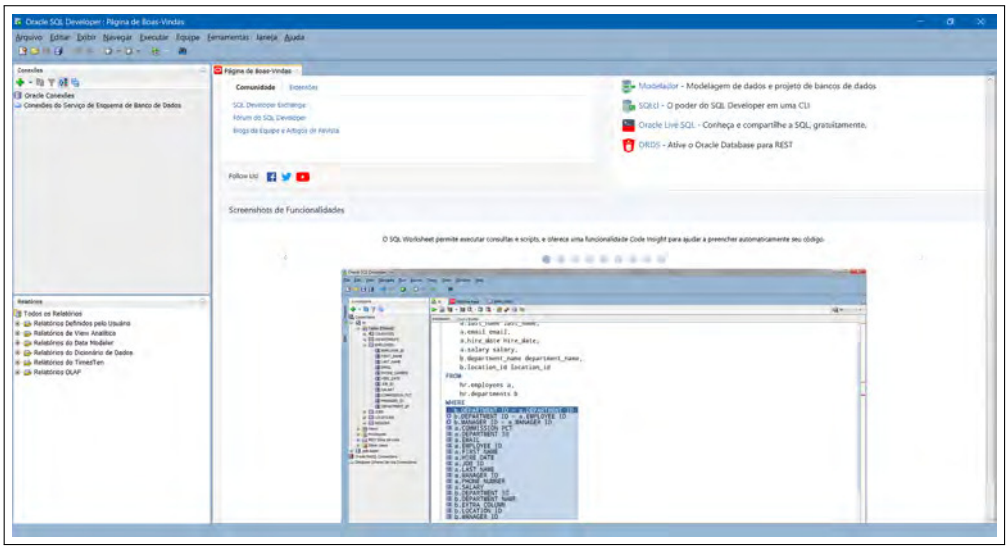
4.2 ORACLE SQL DEVELOPER

O Oracle SQL Developer é um software gratuito para banco de dados, cuja versão gráfica do SQL*Plus oferece aos desenvolvedores uma maneira conveniente de executar tarefas básicas 5. Essa base de dados entrega opções de navegação, criação, edição e exclusão (soltar); execução de instruções e scripts SQL; edição e depuração de código PL / SQL; manipulação e exportação de dados; e visualização e criação de relatórios (MURRAY, 2021).

Utiliza-se esta base de dados para se conectar aos esquemas de banco de dados do BRM usando a autenticação de banco de dados Oracle padrão. Então, uma vez conectado, pode-se realizar operações em objetos no banco de dados.

Todavia, esse software não fica restrita apenas a esquemas Oracle, mas ainda a esquemas para MySQL e bancos de dados de terceiros (não Oracle) selecionados, como Microsoft SQL Server, Sybase Adaptive Server e IBM DB2, e visualizar dados e metadados; e é possível optar pela migração desses esquemas para o banco de dados Oracle.

Figura 5 – Tela Inicial do SQL Developer



Fonte: Elaborado pelo autor, 2021

4.3 RED HAT OPENSHIFT

O Red Hat OpenShift é uma plataforma empresarial de Kubernetes, criada pela Red Hat tendo seu lançamento em 2011. Ela é importante por incorporar a mesma tecnologia que serve como motor para telecomunicações massivas, streaming de vídeo, jogos, serviços bancários e outros formulários. Sua implementação em tecnologias abertas da Red Hat permite que você estenda seu conteúdo em contêineres aplicativos além de uma única nuvem para ambientes no local e com várias nuvens, (RED HAT, 2021).

Kubernetes nada mais é do que uma plataforma em rápida evolução que gerencia aplicativos baseados em contêiner e seus componentes de rede e armazenamento associados. Este foca nas cargas de trabalho de aplicativos, não na infraestrutura dos componentes subjacentes. Essa plataforma fornece uma abordagem declarativa para implementações, apoiada por um conjunto robusto de APIs para operações de gerenciamento. Podendo criar e executar aplicativos modernos, portáteis e baseados em microsserviços usando o Kubernetes para orquestrar e gerenciar a disponibilidade dos componentes de aplicativo, (RED HAT, 2021).

Essa tecnologia de Kubernetes foi criada para uma estratégia de nuvem híbrida aberta 6. Ele integra o OpenShift Container Platform que é a plataforma da Red Hat para desenvolver e executar aplicativos em contêineres. Esta foi projetada para permitir que os aplicativos e os data centers que os suportam se expandam de apenas algumas máquinas e aplicativos para milhares de máquinas que atendem a milhões de clientes.

Os contêineres usam sistemas operacionais Linux pequenos e dedicados sem kernel. Seu sistema de arquivos, rede, cgroups, tabelas de processos e namespaces separados do sistema host, mas os contêineres podem se integrar aos hosts perfeitamente quando necessário.

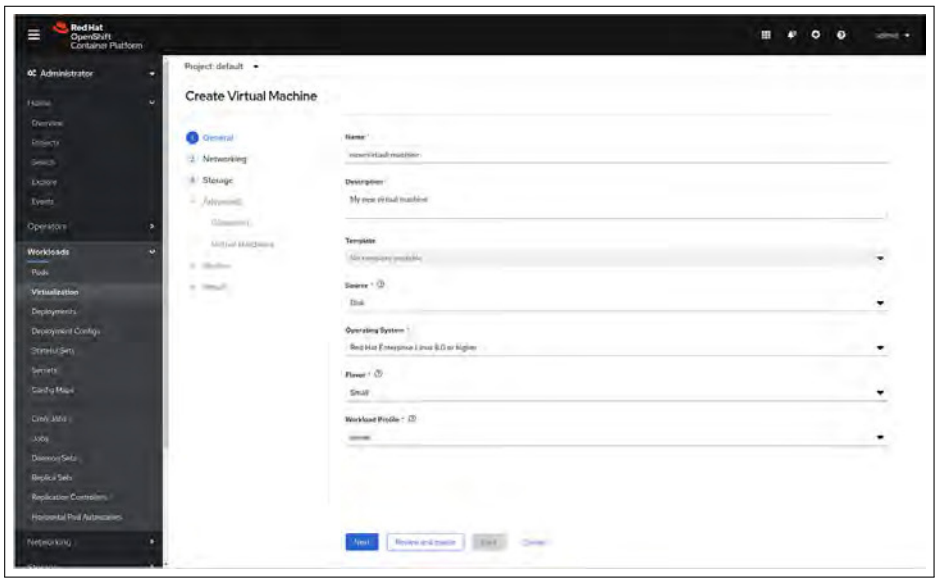
Navegando pelo conceito geral de Kubernetes, (RED HAT, 2021), que é bastante simples, temos:

- Opção de começar com um ou mais nós de trabalho para serem executadas as cargas de trabalho do contêiner.
- Gerenciamento da implantação dessas cargas de trabalho de um ou mais nós do plano de controle (também com nome de nós mestres).
- Envolvimento dos contêineres em uma unidade de implantação chamada pod. O uso de pods fornece metadados extras com o contêiner e oferece a capacidade

de agrupar vários contêineres em uma única entidade de implantação.

- Crie tipos especiais de ativos. Por exemplo, os serviços são representados por um conjunto de pods e uma política própria que define como eles são acessados. Esta política permite que os contêineres se conectem aos serviços de que precisam, mesmo que não tenham os endereços IP específicos para os serviços. Os controladores de replicação são outro ativo especial que indica quantas réplicas de pod são necessários para ser executado por vez. Você pode usar esse recurso para dimensionar automaticamente seu aplicativo para adaptar-se à sua demanda atual.

Figura 6 – Tela do OpenShift



Fonte: Site da Red Hat, 2021

O Red Hat OpenShift está disponível como um serviço de nuvem totalmente gerenciado em nuvens públicas líderes ou como uma oferta de software autogerenciado para organizações que precisam de mais personalização.

4.4 MICROSOFT AZURE DEVOPS

O Azure DevOps é uma plataforma que fornece diversos serviços empresariais para desenvolvedores e equipes de suporte para planejar o trabalho, colaborar no desenvolvimento do código e criar e implantar aplicativos. Ela dá suporte a uma cultura e a um conjunto de processos que trazem desenvolvedores, gerentes de projeto e colaboradores para concluir o desenvolvimento de software (MICROSOFT, 2021).

O Azure DevOps fornece recursos integrados que se pode acessar por meio de seu navegador da Web ou cliente IDE, conforme apresentado na Figura 7.

Figura 7 – Tela do Azure DevOps

Fonte: Site do Azure DevOps, 2021

A plataforma é baseada nas necessidades de negócios podendo contratar todos, um ou mais dos seguintes serviços autônomos (MICROSOFT, 2021):

- O *Azure Repos* fornece repositórios Git ou controle de versão do Team Foundation (TFVC) para o controle do código-fonte de seus códigos.
- O *Azure Pipelines* fornece serviços de compilação e versão para dar suporte à integração e à entrega contínuas de seus aplicativos.
- O *Azure Boards* fornece um pacote de ferramentas Agile para dar suporte ao planejamento e acompanhamento de trabalho, defeitos de código e problemas usando os métodos Kanban e Scrum.
- O *Azure Test Plans* fornece várias ferramentas para testar seus aplicativos, incluindo testes manuais/exploratórios e testes contínuos.
- O *Azure Artifacts* permite que as equipes compartilhem pacotes como Maven, NPM, NuGete muito mais de fontes públicas e privadas e integre o compartilhamento de pacotes em seus pipelines.

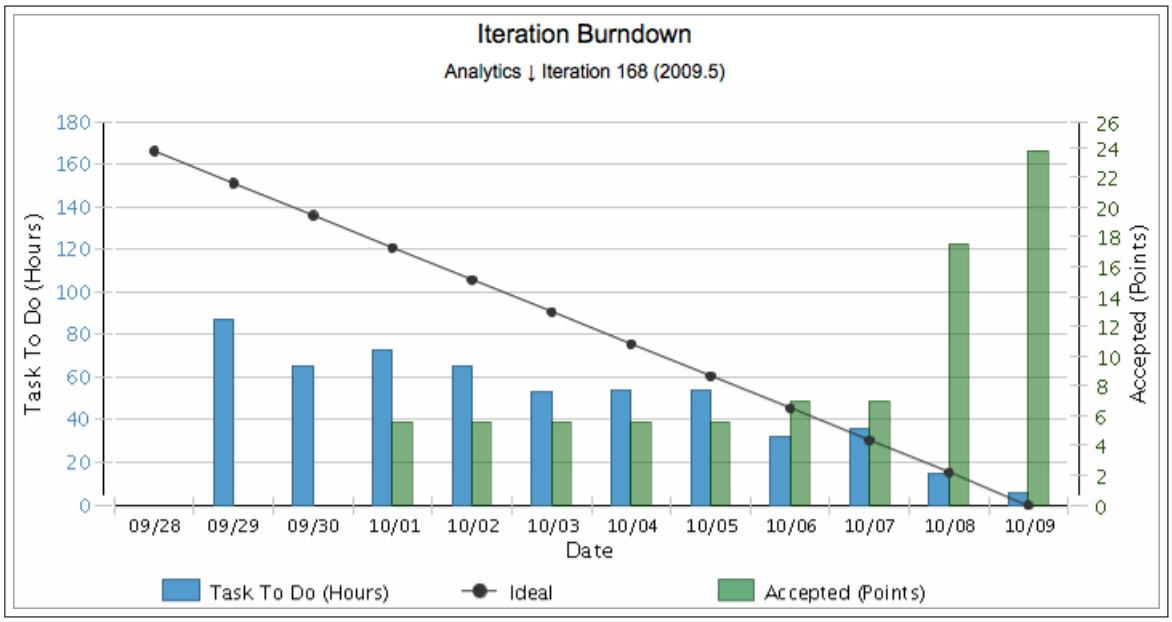
O Azure DevOps dá suporte a adição de extensões e à integração com outros serviços populares, como: Campfire, margem de atraso, Trello, UserVoice e muito mais, e desenvolver extensões personalizadas. Essa plataforma ainda dá suporte à integração com repositórios do GitHub.com e do GitHub Enterprise Server.

5 ATIVIDADES REALIZADAS

A Accenture do Brasil é uma empresa multinacional muito dinâmica e permite que seus funcionários e colaboradores migrem de um projeto para outro, mediante a justificativa para que seja feita essa mudança. O colaborador possui liberdade para desempenhar um papel diferente no processo de desenvolvimento, basta fazer esse alinhamento com o SM do projeto. Até o momento da publicação deste trabalho, cerca de 80% de todo o esforço tem sido direcionado pela empresa aos projetos que usam da tecnologia de BRM.

Em projetos ágeis, a empresa e o time envolvidos precisam fazer todo o mapeamento, não só o acompanhamento, como também os pontos críticos e possíveis impedimentos que possam ocorrer durante o processo de desenvolvimento. O gráfico *Burndown*, figura 8, mostra a quantidade de trabalho restante naquela *sprint* e o número de histórias de usuário aceitas. Com base no qual se pode analisar facilmente se o trabalho da *sprint* está seguindo no caminho certo ou não.

Figura 8 – Gráfico de *Burndown do Rally*



Fonte: *Site* da ferramenta *Rally*, 2021.

Para ter um controle de todo esse desenvolvimento, das Funcionalidades (*Features*), bem como as horas gastas nas Histórias de Usuário (US), nos Testes Unitários (TU) e Testes de Histórias (TS), a empresa faz uso de uma ferramenta chamada *Rally*. A grande vantagem de se utilizar uma ferramenta de gestão ágil é que

muitos dos *feedbacks* podem ser vistos diretamente da plataforma, como por exemplo, quando uma história for aceita ou, em caso contrário, ter acesso ao *feedback* do PO.

A desenvolvedora desse *software, a Rally Software*, capacita as empresas a mudar e agregar valor rapidamente, fornecendo visibilidade em tempo real, em escala corporativa de projetos e métricas de desempenho que ajudam as equipes a evoluir continuamente (BROADCOM, 2021). Dentro dessa visão, o *Rally* surge como uma plataforma empresarial construída com o propósito de escalar as práticas de desenvolvimento ágil.

Partindo para uma visão mais específica da tecnologia, vale salientar que o projeto utiliza uma versão do BRM que é hospedada e executada na nuvem. E, até o presente trabalho, existem os seguintes módulos (repositórios mapeados dentro do Azure DevOps) que merecem uma atenção prévia, são estes (ORACLE, 2020):

- ***BRM Apps:*** Como o próprio nome sugere, é o módulo onde estão todas as funcionalidades desenvolvidas, aqui estão contidos *Shell Scripts* e os arquivos escritos em C. Os *Shell Scripts* (sh) aqui incluídos são desenvolvidos voltados para ambientes Unix.
- ***Configurator:*** É o módulo que contém as configurações do BRM, bem como criação de tabelas e inserção de dados no banco (*database*), aqui são gerenciadas as classes associadas às suas respectivas tabelas. Os arquivos de criaçõ são os PODLs e os de inserção, atualização ou exclusão são os NAPs.
- ***Connection Manager:*** Neste módulo estão configuradas todas as partes relativas as conexões externas, como chaves de acesso, *token* de sessão para APIs, endereços de servidor, porta e outras configurações.
- ***Shared Libs:*** Como o próprio nome sugere, aqui ficam todas as bibliotecas e variáveis que serão utilizadas e compartilhadas entre os módulos do BRM. Quando foram criadas todas as colunas da tabela no banco de dados, os nomes são associados às variáveis que aqui precisam ser previamente declaradas, suas nomenclaturas e instruções de configuração seguem o padrão de acordo com a documentação da Oracle.
- ***HelmChart:*** Esse módulo crítico está associado ao ECE que executa carregamento *on-line* e carregamento *off-line*, ou seja, é uma forma de *backup* permanente do cache no caso de falha de um nó, perda de partição ou assim por diante. O ECE recupera automaticamente os dados do cache do banco de dados de persistência, quando necessário. Aqui também se encontram os dados que são

sincronizados ou recebidos do BRM e, outros, como saldo, histórico de recargas, histórico de pacote recorrente, eventos avaliados e IDs de objetos do Portal (POIDs). Os arquivos aqui contidos também seguem uma estrutura própria de acordo com a documentação da Oracle.

- ***Web Service Manager:*** Aqui estão contidos todos os serviços de integração e comunicação do BRM, usa-se o *Restful* como arquitetura, de modo que o BRM possa realizar uma chamada para um serviço de um outro sistema, a fim de obter informações.

5.1 MODELAGEM DE DADOS

O *Rally* é uma ferramenta utilizada no cotidiano do projeto para gerenciamento dos desenvolvimentos ágeis, aqui também é postado um detalhamento do que é escrito pelo PO e, a parte mais técnica pelo time de arquitetura do BRM. Na figura 9 é demonstrado como é descrito o detalhamento do que foi pedido para esta história.

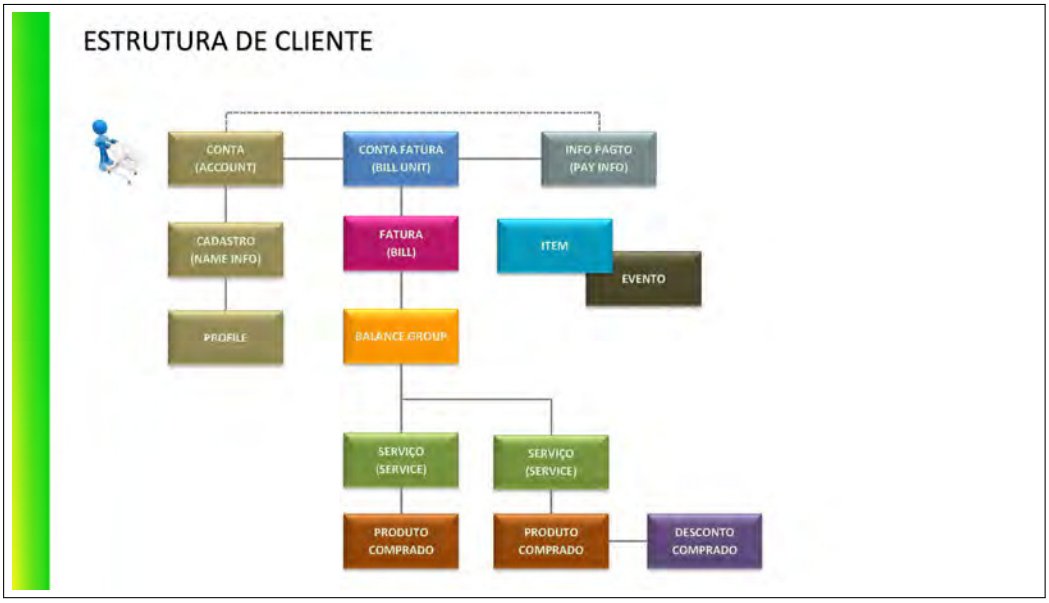
Figura 9 – Detalhamento da US

Descrição	Como sistema de billing gostaria de atualizar o modelo de dados do BRM com as tabelas customizadas criadas no BRM
Critérios de Aceitação	Atualizar o modelo de dados do BRM com as tabelas customizadas criadas no BRM
Tecnologias Envolvidas	BRM
Detalhamento Técnico	Atualizar o modelo de dados do BRM com as tabelas customizadas criadas no BRM referentes ao escopo da squad Faturamento 2 e squad Antecipação – Piloto.
Dependências Externas Funcionais	
Dependências Externas Técnicas	
Requisitos Não Funcionais (NFRs)	

Fonte: Rally, 2021.

Após a leitura do detalhamento técnico na figura 9, toda a coleta de informações prévias, tanto do time de Faturamento II quanto do time de Antecipação, para este último, como essa equipe não existia mais, precisou-se realizar a coleta com um dos arquitetos do projeto. A validação precisava ser feita juntamente com o PO e o SM, visto que era apenas uma atualização da versão 1.1 que pode ser observada, de forma parcial, na figura 10.

Figura 10 – Versão 1.1 da Modelagem

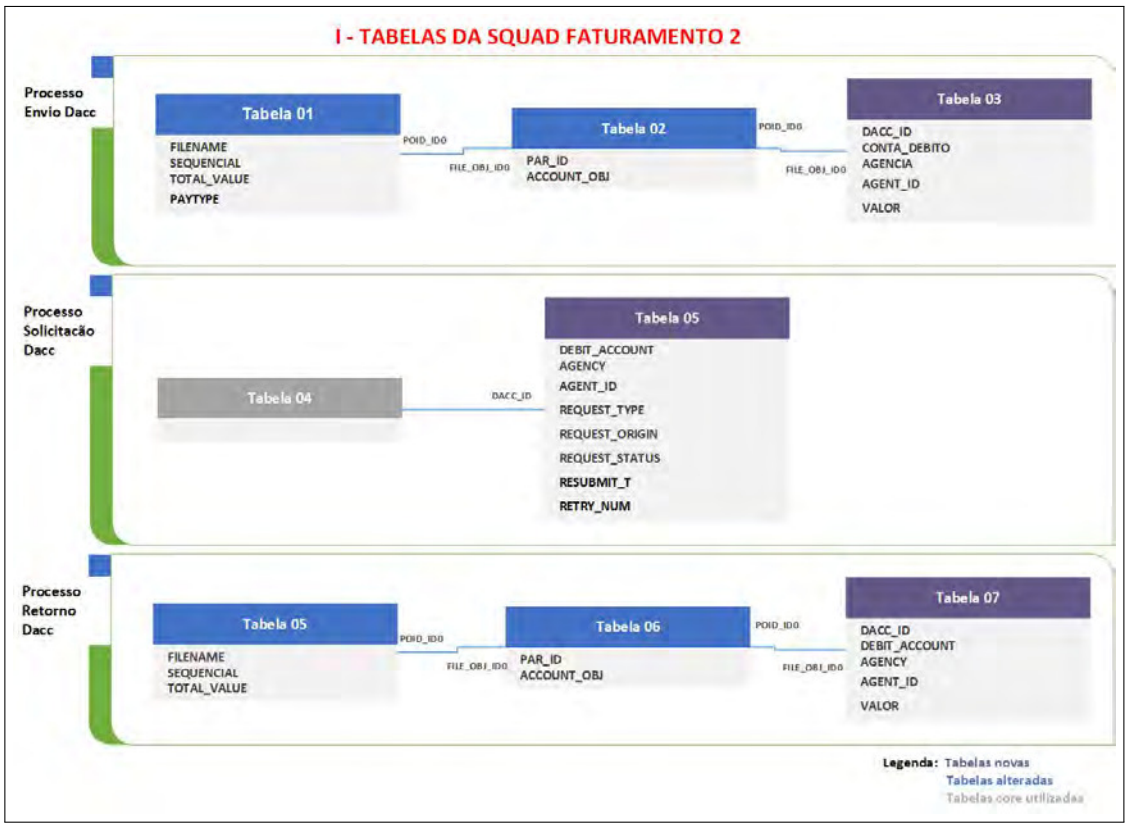


Fonte: Projeto, 2021.

A modelagem acima é uma parcela de como estão organizadas as classes do projeto. Nesse caso, podemos ver uma boa parte da estrutura núcleo *(core)* do BRM, na parte do cliente: temos a conta (account), o cadastro (nameinfo) e o perfil desse cliente (profile). Esse cliente realiza uma compra (bill unit), é gerada uma fatura para essa compra (bill) e temos um grupo de balanço (balance group) que reúne informações dos serviços oferecidos (service), com os produtos comprados juntamente com a oferta/desconto. A classe de itens e eventos estão ligadas aos dados de pagamento (pay info).

Para o desenvolvimento da modelagem foi utilizado um Esquema de Banco de Dados parecido com o demonstrado acima, bem como quais relacionamentos com as tabelas já existentes. Todo o mapeamento resultou em trinta e uma tabelas, excluindo da contagem as tabelas *core* que foram utilizadas, visto que não houve quaisquer alterações nestas; todavia, foram citadas para fins de esclarecimento sobre o fluxo de dados. Vale ressaltar que foi transcrita apenas uma pequena parcela da modelagem.

Figura 11 – Tabelas mapeadas sobre fluxos de DACC



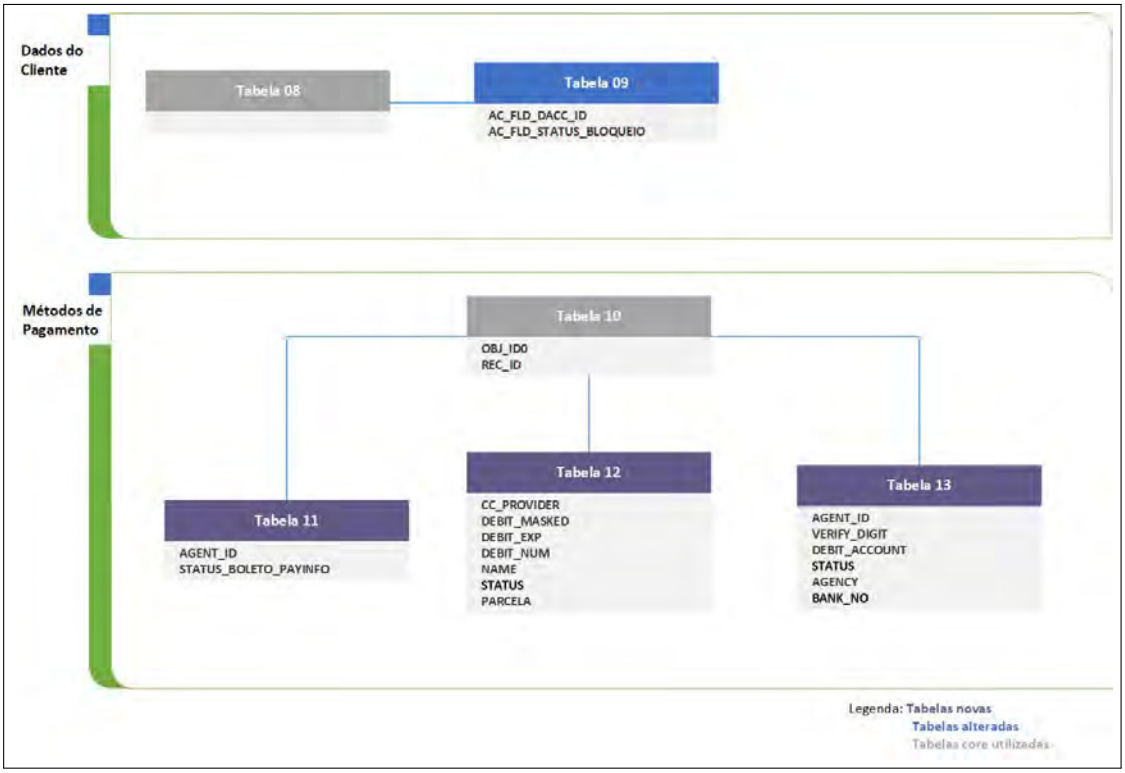
Fonte: Próprio autor, 2021.

As tabelas expostas na imagem 11 são referentes ao processo de Débito Automático em Conta Corrente (DACC) tanto para processo de Envio quanto o de Solicitação e, por fim, o de Retorno; aqui é demonstrado o fluxo associado a cada processo para que melhorasse a compreensão, se comparado a versão antiga da modelagem apresentada na figura 10.

Na figura 12 vemos que o fluxo para os Métodos de Pagamento foi atualizado com três novas tabelas, enquanto no fluxo de Dados de Cliente houve apenas atualização de uma das tabelas do fluxo. Todos os atributos descritos são os que foram adicionados, tanto para as tabelas alteradas e quanto para as novas.

Na modelagem final referentes as tabelas da *squad* de Faturamento 2 foram mapeadas dezoito, sendo onze novas e sete que foram alteradas.

Figura 12 – Tabelas de fluxo do Cliente e Pagamento

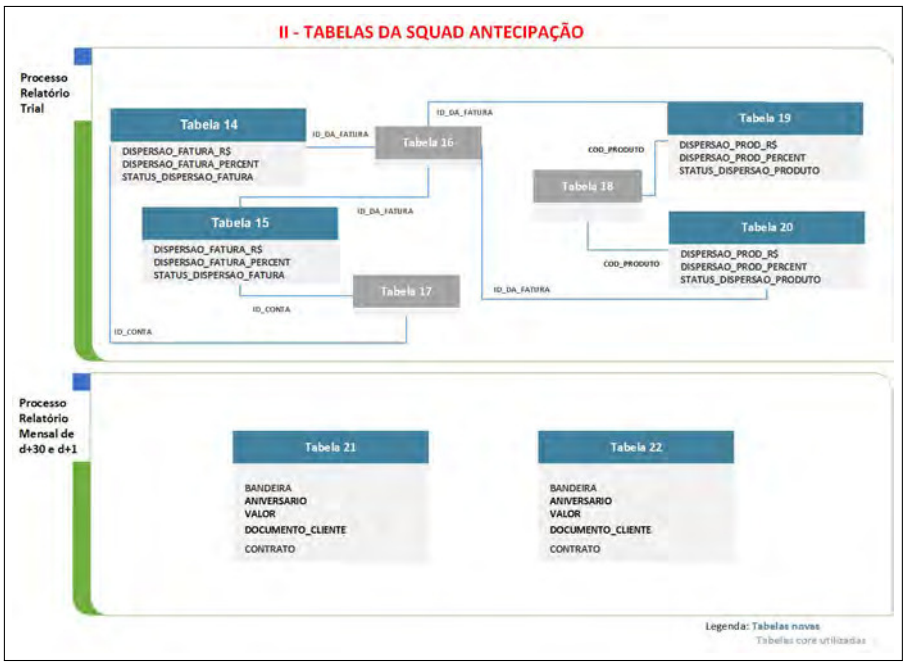


Fonte: Prório autor, 2021.

Uma fatura trial são aquelas para o período de 30 dias, relatórios d+1 é para as do início do mês e d+30 é para as que são no fim do mês. Então, no que tange às da equipe de Antecipação, as tabelas expostas na imagem 13, quatro delas são para relatório de faturas trial e as outras duas são referentes aos relatórios diários (d+1) e mensal (d+30). Esses fluxos são, basicamente, tabelas temporárias criadas para guardar todos os dados dos relatórios que serão gerados.

Já na figura 14 temos mais duas tabelas temporárias que foram criadas para gerar os relatórios referentes ao fluxo de conciliação física de débito automático (DACC). Essas tabelas são populadas através de um processo, além de preencher ele precisa truncar cada uma delas a cada nova interação.

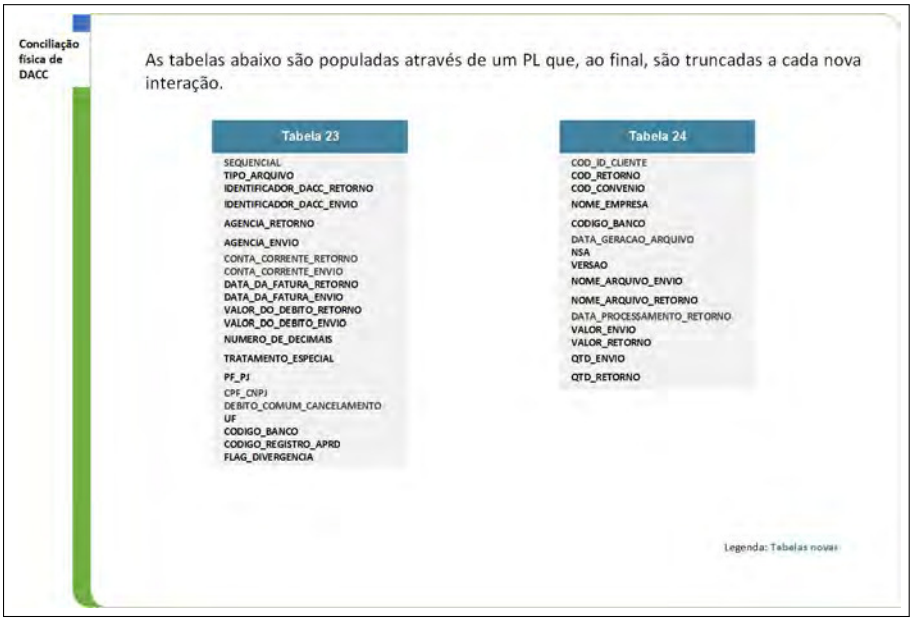
Figura 13 – Tabelas de fluxos de relatório



Fonte: Próprio autor, 2021.

Para essa *squad* foi constatado que foram classes novas implementadas e novos fluxos, que não foram exemplificados aqui por não ter conhecimento aprofundado sobre o que foi desenvolvido por essa equipe, ao final foram um total de treze tabelas mapeadas.

Figura 14 – Tabelas de fluxo de conciliação de DACC



Fonte: Próprio autor, 2021.

Para finalizar a US, a etapa de revisão passou por todo o time e também pelo arquiteto antes da validação que seria feita em forma de reunião com o SM do projeto. Como se tratou de uma modelagem de dados, não houve qualquer necessidade de TU e, consequentemente, não haveria criação de massa para TS. Feita a validação, por parte do SM, e aprovação, o documento foi formalizado e integrado ao arquivo de modelagem já existente.

5.2 SELEÇÃO DE ARQUIVO FISCAL DE ITEM

Como foi apresentado na seção anterior, é utilizado quotidianamente o *Rally* como principal ferramenta para o gerenciamento e detalhamento de histórias. Logo, na figura 15 se encontra o detalhamento daquilo que foi pedido para esta US.

Figura 15 – Detalhamento da US

Descrição	Como sistema de billing gostaria de fazer a seleção das informações necessárias para a geração do arquivo fiscal de Item.
Critérios de Aceitação	O sistema de billing deve selecionar os dados necessários para a geração do arquivo fiscal de Item. Os dados necessários para a seleção estão detalhados no arquivo INDIRETOS - SAIDA SP31 Alguns campos conforme descritos no arquivo são valores fixos ou dependem se uma informação existe ou não para o preenchimento devem ser considerados nessa seleção Os tributos diretos municipais deverão ser enviados do sistema de faturamento para o sistema fiscal, conforme produto ativado para o cliente (UF ativação do cliente). Gerar os arquivos fiscais para os produtos **Produto 01, Produto 02, Produto 03, Produto 04** que tenham a autenticação em UF(UF ativação do cliente) que possuem Regime Especial conforme novo layout mas em nome da (**Empresarial**) - forma consolidada por item. Nos casos do dos Produtos 01 autenticados em **SP/GO** no qual não possuímos o regime especial, a interface deverá ocorrer nota a nota por cliente faturado.
Tecnologias Envolvidas	BRM
Detalhamento Técnico	**BRM** 1) Configurar um novo registro na classe CONF_DATA_EXECUCAO para armazenar o parâmetro de Data de Última execução do processo de Seleção de Item 2) Modificar o processo MTA para Geração de Arquivos Fiscais 2.1) O processo deve iniciar recebendo os parâmetros : * Tipo de Arquivo a ser gerado - Obrigatório, para essa US o valor deverá ser **ITEM** * Data de extração - Opcional, no formato DDMMYYYY apenas quando desejável reprocessamento da data informada 2.2) O processo deve então montar a lista de datas para processamento. * Caso o parâmetro "Data de extração" tenha sido informado, a lista conterá a data informada. * Caso o parâmetro "Data de extração" não tenha sido informado, deverá ser buscada na tabela de Parâmetros de Processos (CONF_DATA_EXECUCAO_T) pelo parâmetro "DtCorteUltimoProcessamento_Item" e montar a lista de datas com todos os dias entre a data informada no parâmetro e a data da execução do processo. 2.3) Para cada data na lista de processamento, o processo irá realizar o seguinte processamento 2.3.1) Selecionar todas os produtos que incidem iss emitidas na data sendo processada 2.3.2) Os campos a serem selecionados devem ser verificados no anexo "INDIRETOS - SAIDA SP31" para layout de Saida de Item (pag 30). 2.3.3) Ordenar a pesquisa por UF 2.3.4) Armazenar os dados na tabela de dados da Interface Fiscal de Item 2.4) Caso o parâmetro "Data das Notas Fiscais" não tenha sido informado, o processo deverá salvar o novo valor do parâmetro "DtCorteUltimoProcessamento_Item" na tabela de Parametros de Processos (CONF_DATA_EXECUCAO_T) com a última data processada. 3) Criar a entidade de dados da Interface Fiscal de Item: FISCAL_ITEM_T

Fonte: Rally, 2021.

Antes de iniciar os desenvolvimentos, o time decidiu confeccionar previamente uma planilha para melhor compreensão sobre cada campo, bem como quais tabelas e

colunas que estarão ligadas na classe resultante de arquivo fiscal; no total, serão sete arquivos fiscais, esta US está relacionada ao segundo. Nessa planilha contém todo o mapeamento para o desenvolvimento de cada arquivo fiscal, dos campos fixos, em branco e variáveis.

A planilha mencionada anteriormente não está transcrita neste documento. Também não houve participação em US do primeiro arquivo fiscal e, até o presente houve atuação no desenvolvimento de cinco deles. Todavia foram escolhidas apenas duas dessas histórias, pois as três outras US de arquivo fiscal são semelhantes as detalhadas nesse documento.

Feita a leitura, entendimento do detalhamento na figura 15, foi feita a coleta de informações prévias. Para esse desenvolvimento serão necessárias alterações em três módulos do BRM respectivamente: ***SL, CM e Configurator***. A sequência de desenvolvimento foi diferente da de submissão para o ambiente.

A primeira parte do desenvolvimento consistiu em alterar o terceiro módulo, o *Configurator*, onde foi necessário criar um arquivo PODL que é responsável por criar uma classe nova que estará associada a uma tabela, a esta foi denominada de */fiscal_item* (demonstração no algoritmo disponível no apêndice A). Nesta classe nova serão mapeados apenas os campos variáveis que precisam ter seus dados guardados no banco.

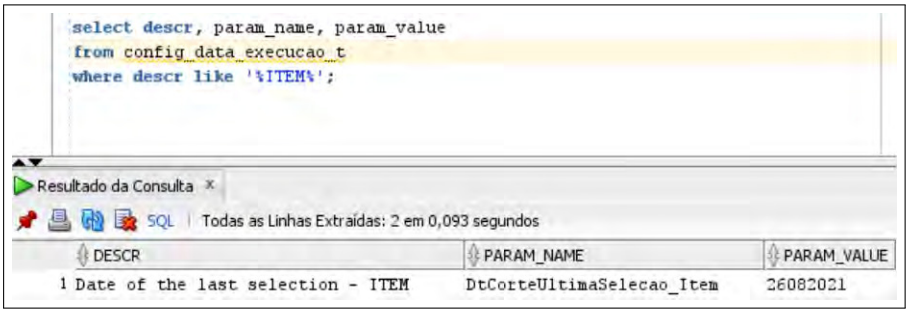
Algoritmo 1 – NAP que popula a tabela de Data Execução

```
1  r << END 1
2  0 PIN_FLD_POID                POID [0] 0.0.0.1 /config/data_execucao
     -1 1
3  0 PIN_FLD_TEMPLATE             STR [0] "update X for /config/
     data_execucao where F1 like V1"
4  0 PIN_FLD_FLAGS                INT [0]  512
5  0 PIN_FLD_ARGS               ARRAY [1]
6  1     PIN_FLD_POID             POID [0] 0.0.0.1 /config/data_execucao
     -1 1
7  0 PIN_FLD_VALUES             ARRAY [0]
8  1     DATA_EXECUCAO   ARRAY [2]
9  2       PARAM_NAME     STR [0] "DtCorteUltimaSelecao_Item"
10 2       MODULE_NAME    STR [0] "mta_selecao_fiscal"
11 2       DESCR          STR [0] "Date of the last selection - ITEM"
12 2       PARAM_VALUE    STR [0] "01011970"
13 END
14 xop PCM_OP_BULK_WRITE_FLDS 32 1
```

Fonte: Elaborado pelo autor

Ainda no módulo do *Configurator* foi criado um NAP para popular uma tabela já existente, como podemos observar no codigo 1 nas linhas 1 e 3 a tabela em questão é a *config_data_execucao* cuja classe associada é a */config/data_execucao*, que iria preencher os campos mapeados nas linhas 8 a 12, aqui queremos armazenar a data atualizada da última execução do processo de seleção dos arquivos fiscais de item, estes que irão popular a tabela *fiscal_item_t* (demonstrado no código disponível no anexo 01); vale destacar que o campo *Param_value* é populado com uma data pré-configurada; na imagem 16 uma *screenshot* parcial da tabela no banco após rodar o processo (PODL e logo a seguir o NAP).

Figura 16 – Captura da tabela de Data Execução



Fonte: Próprio autor, 2021.

Segundo, precisa-se declarar novas variáveis globais, caso não exista, no módulo do ***SL e no CM***, como demonstra o algoritmo 2, o BRM segue o padrão próprio para declarações onde colocamos FLD antes do nome da variável para associar que se trada de um Campo *(Field)*. Então temos *ID_ITEM, VALOR_UNITARIO, LOTE, ITEM_LC*, logo a frente declaramos o tipo de cada uma, no caso, String (FLDT_STR), o *MAKE_FLD* é a função responsável pela criação de novos campos e a numeração são os IDs que cada campo deverá ter.

Algoritmo 2 – Campos novos que foram declarados para o ITEM

```
1 #define FLD_ID_ITEM                    MAKE_FLD(FLDT_STR, 1)
2 #define FLD_VALOR_UNITARIO             MAKE_FLD(FLDT_STR, 2)
3 #define FLD_LOTE                       MAKE_FLD(FLDT_STR, 3)
4 #define FLD_ITEM_LC                    MAKE_FLD(FLDT_STR, 4)
```

Fonte: Elaborado pelo autor

A terceira parte do desenvolvimento consistiu em criar uma função para a seleção dos arquivos de item, aqui mapeamos os campos variáveis que são aqueles que serão guardados no banco de dados (os algoritmos estão disponíveis no apêndice

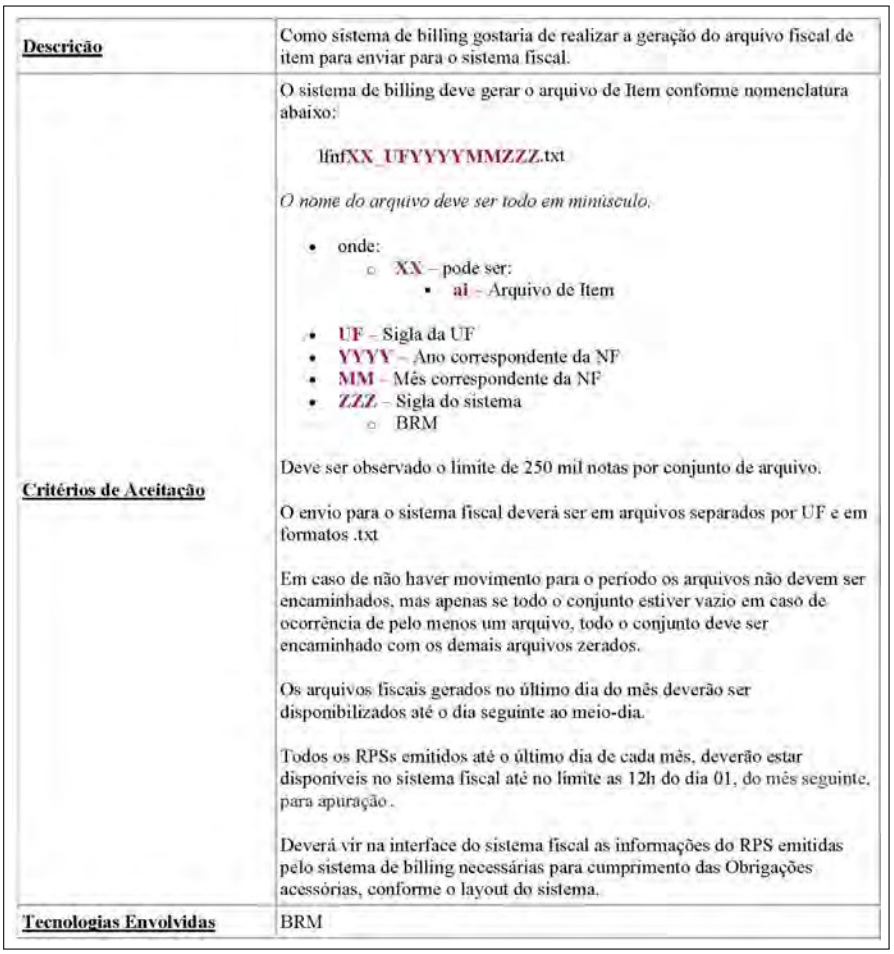
A). Essa etapa é a principal de todo o processo já que é ela quem irá mapear todos os dados disponíveis em outras tabelas para preencher a tabela de item fiscal.

E, por último, realizou-se ainda o desenvolvimento do cabeçalho relativo ao arquivo de item, porém, a versão (demonstrada no código disponível no apêndice A) se trata apenas de uma representação, portanto, não reflete o arquivo original. Esse cabeçalho nada mais é do que um arquivo que contém todos os campos fixos, ou seja, aqueles que não precisam ser guardados em um banco de dados já que as alterações nesses campos serão esporádicas.

5.3 GERAÇÃO DE ARQUIVO FISCAL DE ITEM

Como foi introduzido na seção 5.1, é comumente utilizado o Rally como principal ferramenta para detalhamento dos desenvolvimentos durante a *Sprint*. Na figura 17 é apresentado outro detalhamento, dessa vez que é complemento da US descrita na seção anterior.

Figura 17 – Detalhamento da US



Descrição	Como sistema de billing gostaria de realizar a geração do arquivo fiscal de item para enviar para o sistema fiscal.
Critérios de Aceitação	O sistema de billing deve gerar o arquivo de Item conforme nomenclatura abaixo: lfnfXX_UFYYYYMMZZZ.txt *O nome do arquivo deve ser todo em minúsculo.* • onde: ○ XX – pode ser: ▪ ai – Arquivo de Item • UF – Sigla da UF • YYYY – Ano correspondente da NF • MM – Mês correspondente da NF • ZZZ – Sigla do sistema ○ BRM Deve ser observado o limite de 250 mil notas por conjunto de arquivo. O envio para o sistema fiscal deverá ser em arquivos separados por UF e em formatos .txt Em caso de não haver movimento para o período os arquivos não devem ser encaminhados, mas apenas se todo o conjunto estiver vazio em caso de ocorrência de pelo menos um arquivo, todo o conjunto deve ser encaminhado com os demais arquivos zerados. Os arquivos fiscais gerados no último dia do mês deverão ser disponibilizados até o dia seguinte ao meio-dia. Todos os RPSs emitidos até o último dia de cada mês, deverão estar disponíveis no sistema fiscal até no limite as 12h do dia 01, do mês seguinte, para apuração. Deverá vir na interface do sistema fiscal as informações do RPS emitidas pelo sistema de billing necessárias para cumprimento das Obrigações acessórias, conforme o layout do sistema.
Tecnologias Envolvidas	BRM

Fonte: Rally, 2021

Esta história, especificamente, é complementar à US de seleção, demonstrada no item anterior. Enquanto que na história anterior era necessário criar a tabela que iria armazenar os dados relativos aos arquivos fiscais de item, nesta será necessário gerar esses arquivos fiscais em .TXT já totalmente preenchidos. Como está detalhada na imagem 17, eles têm que seguir uma nomenclatura própria, bem como uma estrutura pré-definida.

Para se ter uma ideia do mapeamento para cada item, ele deverá ter cento e

quarenta campos e com um total de mil e trezentas posições; em outras palavras, caso haja um caractere a mais, menos ou diferente, o arquivo final se torna inconsistente. Por isso, antes da validação dessa US juntamente com o SM e PO, o teste para essa história foi complexo, pois houve a conferência campo a campo do arquivo resultante, da sua nomenclatura e três casos de execução do processo.

Nesta geração foi preciso alterar dois arquivos: um MTA, que contém o processo de geração e, outro que é o *SHELL SCRIPT*, que é a chamada deste e sua validação prévia. No que tange ao desenvolvimento, primeiramente foi feita a alteração do MTA (algoritmos disponíveis no apêndice B), ele já havia sido criado para geração do primeiro arquivo fiscal, portanto, era necessário fazer alterações para que fosse identificado o de ITEM.

Mesmo o arquivo já estando com boa parte do desenvolvimento pronto, com as devidas alterações realizadas, ainda foi necessário fazer diversos ajustes no MTA para que não acontecessem erros de execução, mesmo quando houvesse a chamada para o processo do primeiro arquivo fiscal.

Logo em seguida, foi feita a alteração no *script* que é responsável por validar a chamada do processo antes de fazer sua execução. Esta alteração foi bastante simples, esse SHELL SCRIPT, apresentado no código 3, que é responsável pela validação dos campos passados como parâmetro e também da tabela *Data Execução* antes de encaminhar para o MTA, aqui é demonstrado apenas a parte do script que foi alterada para fazer o reconhecimento do arquivo fiscal de item.

Algoritmo 3 – Script de geração do arquivo de ITEM

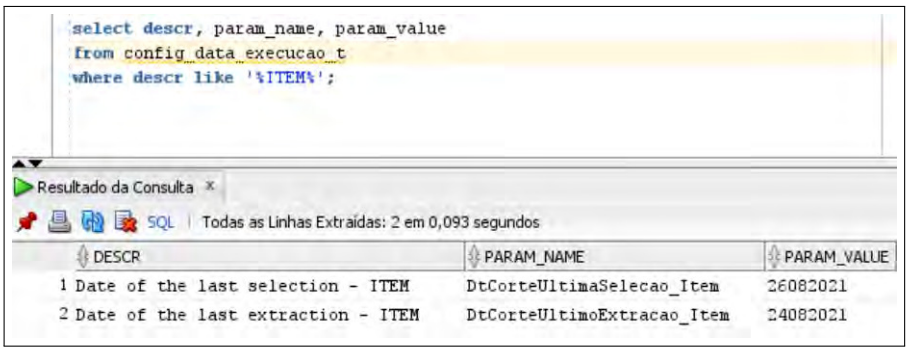
```
function check_param()
{

    if [ -z $1 ];
    then
        echo "[`date +%d/%m/%Y` `date +%H:%M:%S`] ${SCRIPT_NAME}: O
           parametro file_type deve ser informado obrigatoriamente."
        help
    else

        file_type=$1
        echo "[`date +%d/%m/%Y` `date +%H:%M:%S`] ${SCRIPT_NAME}:
           Tipo do arquivo fiscal: $file_type."

        if [ $file_type == "ITEM" ];
        then
            param_name="DtCorteUltimoExtracao_Item";
            table_name="fiscal_item_t";
        else
            help
        fi

    fi
}
```

Fonte: Elaborado pelo autor

Durante o processo de Teste Unitário (TU) ocorreram erros envolvendo as gerações sem passar data como parâmetro, então depois de feitas essas alterações prosseguiu-se para a etapa final.

Na etapa final, foi realizada a validação de todo o processo; na imagem 18 vemos que a tabela contendo os dados da seleção e também a data da extração depois da execução do processo. Uma observação a ser feita é a de que foi criado um novo NAP parecido com o demonstrado no código 1 a diferença é a de que aqui foi feita a declaração para extração.

Figura 18 – Captura da tabela de Data Execução



Fonte: Próprio autor, 2021

Dando prosseguimento à validação, foi feita a execução do *script* passando os seguintes parâmetros: *geracao_iss_fiscal.sh -file_type ITEM*. Os *logs* dessa execução foram guardados para serem anexados no documento de testes. Depois, de volta ao banco de dados foi verificado se houve a alteração na data de execução.

O último passo, a mais demorada e importante das tarefas, que consistiu na validação propriamente dita, de campo a campo, bem como da nomenclatura do arquivo fiscal de item gerado. No código 4, cuja validação foi feita, temos a demonstração do conteúdo do arquivo. O arquivo é preenchido por itens mapeados por localidade (UF), logo, como temos três itens com a UF em GO, o MTA preenche cada linha com um item faturado em Goiás. O nome arquivo recebeu a nomenclatura: *lfnfai_go202110brm*; se observar atentamente ele segue a mesma que foi pedida no detalhamento.

Algoritmo 4 – Arquivo de ITEM gerado

```
1 1         000002BREMPR_EXP_BATM      01UN
  00000000000000100000000000047900000000000479          16082021
                                                  9

  000000000000000
2 2         000002BREMPR_LOTR          01UN
  00000000000000100000000000023900000000000239          17082021
                                                  9

  000000000000000
3 3         000001BREMPR_EXP_HP        01UN
  00000000000000100000000000015900000000000159          17082021
                                                  9

  000000000000000
```

Fonte: Elaborado pelo autor

5.4 DIFICULDADES ENCONTRADAS

A Accenture do Brasil é uma empresa que sempre busca a excelência. Seja no uso das melhores tecnologias ou mesmo aplicando suas melhores práticas, ela se preocupa bastante com o *feedback* fornecido por cada cliente e, no bem-estar de cada colaborador. Ela abriu as portas para a aquisição de um conhecimento técnico e profissional incrivelmente enriquecedor, aprofundando e nivelando o conhecimento de algumas das tecnologias aprendidas durante a graduação, fora dela, bem como em tecnologias emergentes.

A empresa permite ainda que seus times tenham autonomia sobre as soluções apresentadas, sugestão de melhorias nos processos de desenvolvimento, chegando até a auxiliar o time de arquitetura em algumas fases específicas.

Conforme apresentada em sessões anteriores, a tecnologia BRM, fornecida pela Oracle, agrega um sistema completo de cobrança, faturamento e gerenciamento de receitas para provedores de serviço; serviços esses que podem ser derivados de serviços legados ou novas funcionalidades. Em contrapartida, para que um serviço dessa natureza seja desenvolvido, requer uma análise minuciosa da funcionalidade para que ela atenda aos requisitos do cliente, passando primeiramente pelo refinamento dela, para, logo em seguida, ser quebrada em histórias menores e, consequentemente, seu detalhamento técnico se torna crucial.

Por se tratar de uma tecnologia madura, o ponto crítico de se trabalhar com esse tipo de solução é que se precisa de um conhecimento prévio de sua arquitetura, bem como de todo o fluxo e seus respectivos repositórios e, ainda é necessário conhecer pelo menos o básico da linguagem C (*structs*, ponteiros e programação estruturada). Foi uma experiência complicada, a princípio, pois, mesmo assistindo aos vídeos com tutoriais teóricos e práticos sobre a tecnologia, ainda restaram muitas lacunas. Tornou-se uma necessidade que alguns dos outros desenvolvedores auxiliassem e montassem um fluxograma, para que a maior parte dessas dúvidas fossem esclarecidas.

Atrelado a isto, foi preciso ainda investir em um curso mais objetivo na linguagem C para poder relembrar muitos dos conceitos e aplicações que foram estudados durante o segundo período do curso.

Outro ponto de dificuldade encontrada foi a de adaptação aos muitos processos, como: o refinamento do time, onde é necessário que cada história tenha que ser pontuada ou repontuada, dependendo do caso, aprender a pontuar uma US foi uma luta; as *dailys* e as entregas quinzenais, logo no início o aluno foi alocado como

temporário no time, portanto, não podia participar das reuniões diárias (*dailys*) muitas vezes; quando foi escalado para o time, a cobrança nas entregas das histórias se tornou uma meta que deveria ser alcançada e, pelo fato, de não ter podido participar dessas reuniões, o aluno se sentia perdido, sem saber como se portar.

E, por último, uma missão: a de interagir diretamente com pessoas de outras localidades apenas usando o inglês como forma de comunicação primária. O primeiro contato *a priori* foi deveras estranho, o inglês parecia ser engessado; à medida que decorriam os dias e as interações, mesmo que usando artifícios de tradução, estabelecer uma conversa mais direta e com um vocabulário mais polido. Isso acabou por lapidar e fortalecer ainda mais a parte profissional.

Por fim, vale salientar que é relativo o grau de dificuldades, mas que fazem parte da vida pessoal, profissional e acadêmica de cada indivíduo. As aqui apresentadas são atribuídas mais ao preparo inicial que se deu através de materiais pré-prontos, que vão muito além de uma receita de bolo. Embora tenha tido pouco mais de um mês para ter uma assimilação com o BRM e saber os pontos importantes do negócio do cliente, esse tempo foi bastante escasso. Posteriormente, outras pessoas entraram no time oriundos de um treinamento sobre a solução e, enquanto colaborador, não ter participado de um treinamento desse nível, só agravou as muitas lacunas que surgiram durante o desenvolvimento das histórias que foram sendo demandadas.

6 CONSIDERAÇÕES FINAIS

As histórias desenvolvidas no time de Faturamento para a tecnologia de BRM foram de grande aprendizado, pois os clientes solicitam o desenvolvimento de suas soluções personalizadas, a fim de gerarem um alto valor para seus negócios, com base no poder da plataforma da Oracle e nas suas principais tecnologias.

A participação em um projeto dessa magnitude proporcionou experiências diversificadas, permitindo uma grande absorção de conhecimentos. Esta, de fato, alcançou o patamar de experiência profissional enriquecedora, com grandes oportunidades por aprofundar alguns dos principais assuntos que foram ensinados durante a graduação, além de poder reaprender a linguagem de programação estruturada.

Foi possível aplicar conhecimentos no desenvolvimento do projeto, tais como domínio de banco de dados sql, familiaridade com git, conhecimento de estrutura de dados; como foi apresentado na sessão de processos, tecnologias utilizadas e uma sólida base na linguagem C. Além de conhecer a cultura *DevOps*, proporcionou um aprendizado muito grande nos conceitos de conteinerização de sistemas, cujas técnicas foram enriquecedoras de conhecimento, para saber como as aplicações eram desenvolvidas e testadas em ambientes modularizados, bem como a entrega automatizada e constante mapeamento nos versionamentos.

Acredita-se que a tendência das empresas em buscar novos profissionais de Tecnologia da Informação se baseia naqueles que tenham uma constante vontade de adquirir conhecimentos extras e diversificados, não se limitando apenas ao desenvolvimento. Afinal, tecnologias como essa surgem rápido, muitas vezes sem o tempo necessário para que sejam vistas na graduação, todavia isso não se torna um impeditivo, enquanto aluno e profissional somos incentivados a estar sempre se aperfeiçoando e estudando baseado nas tendências do mercado.

Ao termino do trabalho e diante das atividades realizadas, observou-se o enorme aprendizado atrelado ao processo de amadurecimento profissional diante das demandas e desafios que surgem a cada novo planejamento do projeto. É enriquecedor não apenas enquanto profissional, mas também enquanto ser humano já que temos que carregar um DNA de valores e visões não apenas dentro da empresa em que se trabalha, mas fora dela, estas também que um indivíduo carregará desde criança até a fase adulta.

Ciclos fazem parte desse crescimento. Em cada nuance que se atua seja como estagiário ou mesmo *freelancer* pode-se enxergar a relevância de cada etapa para a construção do currículo profissional. E, o quanto o trabalho ali realizado foi importante para a construção do seu aprendizado.

O curso Superior de Tecnologia em Análise e Desenvolvimento de Sistema do Instituto Federal de Ciências e Tecnologia do *Campus* de Cajazeiras se tornou sinônimo de excelência, pois consumou a lapidação dos conhecimentos em algumas tecnologias já utilizadas no desempenho desta profissão, levando a acreditar que toda fórmula de conhecimento que foi disseminado pela instituição auxilia grandemente na capacitação de seus alunos para enfrentar os mais diversos desafios propostos pelo mercado de trabalho.

REFERÊNCIAS

ACCENTURE. **Accenture | Brasil| Que venha a mudança**. United States, 2021. Disponível em: <https://www.accenture.com/br-pt/about/company-index>.

AUDY, J. **SCRUM 360**: Um guia completo e prático de agilidade. São Paulo, 2015.

BROADCOM INC. **Rally® Software**. United States, 2021. Disponível em: <https://www.broadcom.com/products/software/agile-development/rally-software>.

COSTA, S. S. da. **Aumento do número de vagas no mercado tecnológico em ano de pandemia**. 2021. Disponível em: <https://estaciocearanoticias.wordpress.com/2021/05/05/aumento-do-numero-de-vagas-no-mercado-tecnologico-em-ano-de-pandemia/>.

GOMES, A. F. **AGILE**: Desenvolvimento de software com entregas frequentes e foco no valor de negócio. São Paulo, 2014.

MICROSOFT CORPORATION. **Azure DevOps/Documentação de início**. United States, 2021. Disponível em: <https://docs.microsoft.com/pt-br/azure/devops/get-started/?view=azure-devops>.

MURRAY, A. S. . C. C. . C. **Oracle SQL Developer User's Guide, Release 21.2**. United States, 2021.

ORACLE CORPORATION. **Oracle Communications Billing and Revenue Management Concepts, Release 12.0**. United States, 2020.

RED HAT INC. **OpenShift Container Platform 4.8 Architecture**: An overview of the architecture for openshift container platform. United States, 2021.

SABBAGH, R. **SCRUM**: Gestão Ágil para projetos de sucesso. São Paulo, 2014.

SOMMERVILLE, I. **Engenharia de Software - 10ª edição**. São Paulo, 2019.

APÊNDICE A – IMPLEMENTAÇÕES DA SELEÇÃO DO ITEM FISCAL

PODL é uma extensão de um arquivo de criação de classes e que já faz o mapeamento dos campos no banco de dados. Nos algoritmos 5 ao 7 está demonstrado como se constrói uma classe no BRM, essa estrutura de criação é própria da Oracle, portanto é seguida à risca.

MTA é uma implementação na linguagem C de um script, este arquivo é a própria lógica de negócio. No algoritmo 8 ao 13 há a exemplificação da função principal cujo objetivo é o de selecionar e preencher os campos do item fiscal no banco de dados.

Algoritmo 5 – PODL de criação de classe para ITEM - Parte 1

```
STORABLE CLASS /fiscal_item {

    READ_ACCESS = "Self";
    WRITE_ACCESS = "Self";
    DESCR = "Dados do Arquivo Fiscal de Item";
    IS_PARTITIONED = "0";

    #Fields
    POID ACCOUNT_OBJ {
        DESCR = "Objeto da Conta.";
        ORDER = 0;
        CREATE = Required;
        MODIFY = Writeable;
        AUDITABLE = 0;
        ENCRYPTABLE = 0;
        SERIALIZABLE = 0;
    }

    POID BILL_OBJ {
        DESCR = "Objeto da Fatura.";
        ORDER = 0;
        CREATE = Required;
        MODIFY = Writeable;
        AUDITABLE = 0;
        ENCRYPTABLE = 0;
        SERIALIZABLE = 0;
    }

    STRING NUMERO_SEQUENCIAL {
        DESCR = "Numero do arquivo fiscal Gerado.";
        ORDER = 0;
        LENGTH = 255;
        CREATE = Optional;
        MODIFY = Writeable;
        AUDITABLE = 0;
        ENCRYPTABLE = 0;
        SERIALIZABLE = 0;
    }
```

Algoritmo 6 – PODL de criação de classe para ITEM - Parte 2

```
    STRING ID_ITEM {
        DESCR = "Numero do Item.";
        ORDER = 0;
        LENGTH = 255;
        CREATE = Optional;
        MODIFY = Writeable;
        AUDITABLE = 0;
        ENCRYPTABLE = 0;
        SERIALIZABLE = 0;
    }

    STRING CODIGO_PRODUTO_CHAR {
        DESCR = "Codigo do Produto.";
        ORDER = 0;
        LENGTH = 255;
        CREATE = Optional;
        MODIFY = Writeable;
        AUDITABLE = 0;
        ENCRYPTABLE = 0;
        SERIALIZABLE = 0;
    }

    DECIMAL VALOR {
        DESCR = "Valor unitario do item.";
        ORDER = 0;
        CREATE = Optional;
        MODIFY = Writeable;
        AUDITABLE = 0;
        ENCRYPTABLE = 0;
        SERIALIZABLE = 0;
    }

    DECIMAL VALOR_TOTAL {
        DESCR = "Valor bruto do item.";
        ORDER = 0;
        CREATE = Optional;
        MODIFY = Writeable;
        AUDITABLE = 0;
        ENCRYPTABLE = 0;
        SERIALIZABLE = 0;
    }

    STRING DATA_EMISSAO {
        DESCR = "Data de Emissao da Nota Fiscal.";
        ORDER = 0;
        LENGTH = 255;
        CREATE = Optional;
        MODIFY = Writeable;
        AUDITABLE = 0;
        ENCRYPTABLE = 0;
        SERIALIZABLE = 0;
    }
```

Algoritmo 7 – PODL de criação de classe para ITEM - Parte 3

```
    STRING MUNICIPIO {
        DESCR = "Municipio do Cliente.";
        ORDER = 0;
        LENGTH = 255;
        CREATE = Optional;
        MODIFY = Writeable;
        AUDITABLE = 0;
        ENCRYPTABLE = 0;
        SERIALIZABLE = 0;
    }

    STRING NOME_ARQUIVO {
        DESCR = "Nome do arquivo gerado.";
        ORDER = 0;
        LENGTH = 255;
        CREATE = Optional;
        MODIFY = Writeable;
        AUDITABLE = 0;
        ENCRYPTABLE = 0;
        SERIALIZABLE = 0;
    }
}

STORABLE CLASS /fiscal_item IMPLEMENTATION ORACLE7 {

    SQL_TABLE = "fiscal_item_t";

    #  Fields
    POID ACCOUNT_OBJ {SQL_COLUMN = "account_obj";}
    POID BILL_OBJ {SQL_COLUMN = "bill_obj";}
    STRING NUMERO_SEQUENCIAL {SQL_COLUMN = "num_rps";}
    STRING ID_ITEM {SQL_COLUMN = "id_item";}
    STRING CODIGO_PRODUTO_CHAR {SQL_COLUMN = "cod_produto";}
    DECIMAL VALOR {SQL_COLUMN = "valor_unitario";}
    DECIMAL VALOR_TOTAL {SQL_COLUMN = "valor_bruto";}
    STRING DATA_EMISSAO {SQL_COLUMN = "dt_emissao";}
    STRING MUNICIPIO {SQL_COLUMN = "municipio";}
    STRING NOME_ARQUIVO {SQL_COLUMN = "nome_arquivo";}

}
```

Fonte: Elaborado pelo autor

Algoritmo 8 – MTA seleção de arquivo fiscal de ITEM - Parte 1

```
1  /*******************************************************************
2   * Funcao principal para selecionar os campos de ITEM fiscal
3  *******************************************************************/
4  static int
5  mta_selecao_fiscal_item(pcm_context_t       *ctxp, pin_flist_t      *
       in_flistp, pin_errbuf_t        *ebufp)
6  {
7
8      //Todas as variaveis e ponteiros declarados
9      poid_t              *account_pdp                    = NULL;
10     poid_t              *bill_pdp                       = NULL;
11     poid_t              *service_pdp                    = NULL;
12     char                *num_fiscal_str                 = NULL;
13     char                *id_item_str                    = NULL;
14     char                *cod_produto_str                = NULL;
15     pin_decimal_t       *valorUnitario                  = NULL;
16     pin_decimal_t       *valorBruto                     = NULL;
17     time_t              *dt_emissao                     = NULL;
18     char                *cod_municipio_str              = NULL;
19     pin_flist_t         *out_account_flistp             = NULL;
20     pin_flist_t         *nameinfoAcct_flistp            = NULL;
21     char                *aci_adress_str                 = NULL;
22     pin_flist_t         *out_nota_fiscal_flistp         = NULL;
23     char                *ptrString                      = NULL;
24     char                getPhrase[2056]                 = "";
25     char                szAciAddressTmp[512];
26     int                 count                           = 1;
27     int                 enderecoid                      = 0;
28     int                 *enderecoid_pointer             = NULL;
29     pin_flist_t         *out_produtos_comprados_flistp  = NULL;
30     pin_flist_t         *out_logradouro_flistp          = NULL;
31     char                szNFCreateDTTmp[150]            = "";
32     struct              tm *emissao_t;
33     int                 i_mkt;
34     pin_flist_t         *item_inflistp                  = NULL;
35     pin_flist_t         *item_outflistp                 = NULL;
36     poid_t              *item_pdp                       = NULL;
37     int                 i64_dbNumber                    = 0;
38     int                 process_status          = AC_ERR_SUCCESS;
```

Algoritmo 9 – MTA seleção de arquivo fiscal de ITEM - Parte 2

```
1     if (PIN_ERR_IS_ERR(ebufp)){
2         process_status = AC_ERR_FAILED;
3         goto cleanup;
4     }
5
6     PIN_ERR_CLEAR_ERR(ebufp);
7
8     //Lista de Clientes
9     account_pdp = FLIST_FLD_GET(in_flistp, PIN_FLD_ACCOUNT_OBJ, 0,
          ebufp);
10    bill_pdp = FLIST_FLD_GET(in_flistp, PIN_FLD_BILL_OBJ, 0, ebufp);
11    service_pdp = FLIST_FLD_GET(in_flistp, PIN_FLD_SERVICE_OBJ, 0,
          ebufp);
12
13    //Captando informacoes relevantes da classe /item
14    num_rps_str = FLIST_FLD_GET(in_flistp, NUMERO_SEQUENCIAL, 0,
          ebufp);
15    id_item_str = FLIST_FLD_GET(in_flistp, ID_ITEM, 0, ebufp);
16    cod_produto_str = FLIST_FLD_GET(in_flistp, CODIGO_PRODUTO_CHAR,
          0, ebufp);
17    valorUnitario = FLIST_FLD_GET(in_flistp, VALOR, 0, ebufp);
18    valorBruto = FLIST_FLD_GET(in_flistp, VALOR_TOTAL, 0, ebufp);
19    dt_emissao = FLIST_FLD_GET(in_flistp, DATA_EMISSAO, 0, ebufp);
20    cod_municipio_str = FLIST_FLD_GET(in_flistp, MUNICIPIO, 0, ebufp)
          ;
21
22    /*******************************
23    * Lendo o objeto /account
24    *******************************/
25    out_account_flistp = mta_selecao_fiscal_get_account_info (ctxp,
          account_pdp, ebufp);
26    if (ERR_IS_ERR(ebufp)){
27        ERR_LOG_MSG(ERR_LEVEL_ERROR, "LOGS: account search error");
28        process_status = ERR_FAILED;
29        goto cleanup;
30    }
31
32    //Captando informacoes relevantes da classe /account
33    nameinfoAcct_flistp = FLIST_ELEM_GET(out_account_flistp,
          PIN_FLD_NAMEINFO, 0, 0, ebufp);
34    aci_adress_str = FLIST_FLD_GET(nameinfoAcct_flistp,
          PIN_FLD_ADDRESS, 0, ebufp);
```

Algoritmo 10 – MTA seleção de arquivo fiscal de ITEM - Parte 3

```
1     /*******************************
2     * Lendo o objeto /nota_fiscal
3     *******************************/
4     out_nota_fiscal_flistp = mta_selecao_fiscal_get_nota_fiscal_info
          (ctxp, account_pdp, bill_pdp, ebufp);
5
6     if (ERR_IS_ERR(ebufp)){
7         ERR_LOG_MSG(ERR_LEVEL_ERROR, "AC LOGS: account search error")
              ;
8         process_status = ERR_FAILED;
9         goto cleanup;
10    }
11
12    //Captando informacoes relevantes da classe /nota_fiscal
13    num_fiscal_str = FLIST_FLD_GET(out_nota_fiscal_flistp, ,
          NUMERO_SEQUENCIAL, ebufp);
14    dt_emissao = FLIST_FLD_GET(out_nota_fiscal_flistp, DATA_EMISSAO,
          0, ebufp);
15
16    /*************************************
17    * Lendo o objeto /produtos_comprados
18    *************************************/
19    out_produtos_comprados_flistp =
          mta_selecao_fiscal_get_produtos_comprados_info (ctxp,
          account_pdp, service_pdp, ebufp);
20
21    if (ERR_IS_ERR(ebufp)){
22        ERR_LOG_MSG(ERR_LEVEL_ERROR, "LOGS: account search error");
23        process_status = ERR_FAILED;
24        goto cleanup;
25    }
26
27    //Captando informacoes relevantes da classe /produtos_comprados
28    cod_produto_str = FLIST_FLD_GET(out_purchased_product_flistp,
          PRODUCT_ID, 0, ebufp);
```

Algoritmo 11 – MTA seleção de arquivo fiscal de ITEM - Parte 4

```
1     /*************************************************
2     * Pegando o ID dos enderacos dos clientes
3     *************************************************/
4     ERR_LOG_MSG(ERR_LEVEL_DEBUG, "LOGS: Retrieving data from address
         field");
5
6     strcpy(getPhrase, aci_adress_str);
7     ptrString = strtok(getPhrase, "|");
8
9     while (ptrString != NULL)
10    {
11        if (ptrString != NULL)
12        {
13            if (count == 1) {
14                enderecoid = atoi((char*)ptrString);
15                enderecoid_pointer = (int*)enderecoid;
16                ERR_LOG_MSG(ERR_LEVEL_DEBUG, "Parsing ENDERICO ID: ")
                     ;
17                ERR_LOG_MSG(ERR_LEVEL_DEBUG, ptrString);
18            }
19
20            if (count == 6) {
21                strcpy(szAciAddressTmp, ptrString);
22                ERR_LOG_MSG(ERR_LEVEL_DEBUG, "Parsing COMPLEMENTO: ")
                     ;
23                ERR_LOG_MSG(ERR_LEVEL_DEBUG, szAciAddressTmp);
24            }
25
26            if (count > 6) {
27                strcat(szAciAddressTmp, " ");
28                strcat(szAciAddressTmp, ptrString);
29                ERR_LOG_MSG(ERR_LEVEL_DEBUG, "Parsing COMPLEMENTO: ")
                     ;
30                ERR_LOG_MSG(ERR_LEVEL_DEBUG, szAciAddressTmp);
31            }
32        }
33        ptrString = strtok(NULL, "|");
34        count++;
35    } //end while
36
37    free(ptrString);
38
39    /*********************************
40    * Leitura do objeto /logradouros
41    *********************************/
42    out_logradouro_flistp = mta_selecao_fiscal_get_logradouro_info (
         ctxp, account_pdp, (int *) enderecoid, ebufp);
```

Algoritmo 12 – MTA seleção de arquivo fiscal de ITEM - Parte 5

```
1     if (ERR_IS_ERR(ebufp)){
2         ERR_LOG_MSG(ERR_LEVEL_ERROR, "LOGS: tmp logradouro search
              error");
3         process_status = ERR_FAILED;
4         goto cleanup;
5     }
6
7     //Pegando informacoes relevantes da classe /logradouros
8     ERR_LOG_MSG(ERR_LEVEL_DEBUG, "LOGS: Retrieving data from /
          logradouros object");
9
10    cod_municipio_str = FLIST_FLD_GET(out_logradouro_flistp, CNL, 0,
          ebufp);
11
12    /*****************************************
13    * Preparando os campos de data
14    *****************************************/
15    ERR_LOG_MSG(ERR_LEVEL_DEBUG, "AC LOGS: Preparing data fields:
          DATA_EMISSAO");
16
17    //Formulacao da variavel AC_FLD_NF_CREATE_DT
18    emissao_t =  localtime(dt_emissao);
19    i_mkt = mktime(emissao_t);
20    strftime(szNFCreateDTTmp, 10, "%d%m%Y", emissao_t);
21
22    if (ERR_IS_ERR(ebufp)){
23        ERR_LOG_MSG(ERR_LEVEL_ERROR, "LOGS: setting date fields");
24        process_status = ERR_FAILED;
25        goto cleanup;
26    }
27
28    /*****************************************
29    * Criacao de um novo objeto /fiscal_item
30    *****************************************/
31    ERR_LOG_MSG(ERR_LEVEL_DEBUG, "creating fiscal_item object");
32
33    i64_dbNumber = POID_GET_DB(account_pdp);
34
35    itemiss_inflistp = FLIST_CREATE(ebufp);
36    item_pdp = POID_CREATE(i64_dbNumber, "/fiscal_item", -1, ebufp);
37    FLIST_FLD_PUT(item_inflistp, POID, item_pdp, ebufp);
38    FLIST_FLD_SET(item_inflistp, ACCOUNT_OBJ, account_pdp, ebufp);
39    FLIST_FLD_SET(item_inflistp, BILL_OBJ, bill_pdp, ebufp);
40    FLIST_FLD_SET(item_inflistp, NUMERO_SEQUENCIAL, num_rps_str,
          ebufp); //retira da tabela de nota fiscal
41    FLIST_FLD_SET(item_inflistp, ID_ITEM, id_item_str, ebufp); //
          retira da tabela de itens
```

Algoritmo 13 – MTA seleção de arquivo fiscal de ITEM - Parte 6

```
1     FLIST_FLD_SET(item_inflistp, CODIGO_PRODUTO, cod_produto_str,
          ebufp); //retira da tabela de produtos comprados
2     FLIST_FLD_SET(item_inflistp, VALOR, valorUnitario, ebufp); //
          retira da tabela de itens
3     FLIST_FLD_SET(item_inflistp, VALOR_TOTAL, valorBruto, ebufp); //
          retira da tabela de itens
4     FLIST_FLD_SET(item_inflistp, DATA_EMISSAO, szNFCreateDTTmp, ebufp
          ); //retira da tabela de nota fiscal
5     FLIST_FLD_SET(item_inflistp, MUNICIPIO, cod_municipio_str, ebufp)
          ;
6     //retira da tabela de endere os
7     FLIST_FLD_SET(item_inflistp, NOME_ARQUIVO, "", ebufp);
8
9     if (ERR_IS_ERR(ebufp)){
10        ERR_LOG_MSG(ERR_LEVEL_ERROR, "LOGS: fields");
11        process_status = ERR_FAILED;
12        goto cleanup;
13    }
14
15    // Executando PCM_OP_CREATE_OBJ
16    PCM_OP(ctxp, PCM_OP_CREATE_OBJ, 0, item_inflistp, &
          itemiss_outflistp, ebufp);
17    if (ERR_IS_ERR(ebufp)){
18        ERR_LOG_MSG(ERR_LEVEL_ERROR, "AC LOGS: object");
19        process_status = ERR_FAILED;
20        goto cleanup;
21    }
22
23    // Liberando espaco da memoria
24    cleanup:
25        FLIST_DESTROY_EX (&out_account_flistp, NULL);
26        FLIST_DESTROY_EX (&out_nota_fiscal_flistp, NULL);
27        FLIST_DESTROY_EX (&out_produtos_comprados_flistp, NULL);
28        FLIST_DESTROY_EX (&out_logradouro_flistp, NULL);
29        FLIST_DESTROY_EX (&item_inflistp, NULL);
30        FLIST_DESTROY_EX (&item_outflistp, NULL);
31
32    return process_status;
33 }
```

Fonte: Elaborado pelo autor

O cabeçalho é um arquivo em SHELL SCRIPT, geralmente utilizado para realizar funções bastante especifícas. Nesse caso, código 14, ele mapeiará todos os campos fixos do arquivo fiscal.

Algoritmo 14 – Cabeçalho do arquivo de ITEM

```
#ifndef lint
static const char _fiscal_h_Sccs_id[] =
        "@(#)%$Id: fiscal.h 5249 YYYY-MM-DD HH:MM:SSZ $%";
#endif

#ifndef FISCAL_H_
#define FISCAL_H_
#define MAX_LINES 250000

/* Global Define */
#define REGISTRO "RPS"
#define SPACE " "
#define ZERO "0"

/* Arquivo Item */
#define NUM_NF "          "
#define ID_ITEM "      "
#define COD_PRODUTO "                    "
#define COD_STATUS "01"
#define COD_UNID_MEDIDA "UN  "
#define QUANTIDADE "000000000000001"
#define VALOR_UNITARIO "               "
#define VALOR_BRUTO "              "
#define LOTE "          "
#define DATA_EMISSAO "        "
#define DESCRICAO_NOTA "                                                "
#define COD_GRUPO_PRODUTO "9   "
#define TRANSACAO "  "
#define OPERACAO " "
#define CLIENTE " "
#define COD_MUNICIPIO "          "
#define DCR "               "
#define HASH_CODE "                                  "
#define CATEGORIA "    "
#define QTDE_CONTRATADA "0000000000000000"
#define COD_FINALIZA " "

#endif /* AC_FISCAL_H_ */
```

Fonte: Elaborado pelo autor

APÊNDICE B – IMPLEMENTAÇÃO DA GERAÇÃO DO ITEM FISCAL

Diferente do MTA no algoritmo 8 que fazia a seleção e preenchia os campos variáveis no banco de dados, no algoritmo 15 há a exemplificação da função principal de geração cujo objetivo é o de receber os dados do cabeçalho (código 14) juntamente com os campos populados no banco e preencher o arquivo propriamente dito de item fiscal. A validação desse arquivo em extensão de texto pode ser vista no código 4.

Algoritmo 15 – Função de Geração de Arquivo de Item - Parte 1

```
1  static int pin_mta_ac_export_fiscal_file_item(pcm_context_t *ctxp,
       pin_flist_t *srch_res_flistp, pin_flist_t *op_in_flistp,
       pin_errbuf_t *ebufp){
2       // Variaveis auxiliares
3       FILE *fPtr;
4       char bodyFile[4 * BUFSIZ] = "";
5       char file_line[6000] = {""};
6       char temp_buffer[2028] = "";
7       char temp_format[2028] = "";
8       char temp_rem_buffer[2028] = "";
9       char temp_rem_format[2028] = "";
10      char file_name_final[250] = {""};
11      char filename[250] = "";
12      // Campos preenchidos no banco
13      char *num_nf = NULL;
14      char *dt_emissao = NULL; //Data da emisso do arquivo fiscal
15      char *vlr_unitario = NULL;
16      char *vlr_bruto = NULL;
17      pin_decimal_t *vlr_unitario_dec = NULL;
18      pin_decimal_t *vlr_bruto_dec = NULL;
19      // Campos do Cabecalho
20      char *cod_produto = COD_PRODUTO;
21      char *cod_status = COD_STATUS;
22      char *cod_unid_medida = COD_UNID_MEDIDA;
23      char *quantidade = QUANTIDADE;
24      char *lote = LOTE;
25      char *descricao_nota = DESCRICAO_NOTA;
26      char *cod_grp_produto = COD_GRP_PRODUTO;
27      char *tp_transacao = TRANSACAO;
28      char *tp_operacao = OPERACAO;
29      char *tp_cliente = CLIENTE;
30      char *cod_municipio = COD_MUNICIPIO;
31      char *dcr = DCR;
32      char *hash_code = HASH_CODE_ITEM;
33      char *categoria_item = CATEGORIA;
34      char *qtde_contratada = QTDE_CONTRATADA;
35      char *ind_finaliza = IND_FINALIZA;
```

Algoritmo 16 – Função de Geração de Arquivo de Item - Parte 2

```
1     // Preenchimento dos campos
2     num_nf = (char *)FLIST_FLD_GET(srch_res_flistp, NUMERO_SEQUENCIAL
          , 1, ebufp);
3     strcpy(temp_buffer, "\0");
4     strcpy(temp_format, "\0");
5     pin_mta_ac_fiscal_space_filler(temp_format, num_nf, 10);
6     sprintf(temp_buffer, "%-10.10s", temp_format);
7     strcat(file_line, temp_buffer);
8     if (ERR_IS_ERR(ebufp))
9     {
10        ERR_LOG_EBUF(ERR_LEVEL_ERROR, "mta_export_fiscal_item: error
              preparing num_nf", ebufp);
11    }
12
13    strcat(file_line, cod_cfop_legal);
14
15    id_item = (char *)PIN_FLIST_FLD_GET(srch_res_flistp, ID_ITEM, 1,
          ebufp);
16    strcpy(temp_buffer, "\0");
17    strcpy(temp_format, "\0");
18    pin_mta_ac_fiscal_char_filler(id_item, 6, 0, temp_format);
19    sprintf(temp_buffer, "%-6.6s", temp_format);
20    strcat(file_line, temp_buffer);
21    if (PIN_ERR_IS_ERR(ebufp))
22    {
23        PIN_ERR_LOG_EBUF(PIN_ERR_LEVEL_ERROR, "
              pin_mta_ac_export_fiscal_file_item: error preparing id
              item", ebufp);
24    }
25
26    cod_produto = (char *)FLIST_FLD_GET(srch_res_flistp,
          CODIGO_PRODUTO_CHAR, 1, ebufp);
27    strcpy(temp_buffer, "\0");
28    strcpy(temp_format, "\0");
29    pin_mta_ac_fiscal_space_filler(temp_format, cod_produto, 20);
30    sprintf(temp_buffer, "%-20.20s", temp_format);
31    strcat(file_line, temp_buffer);
32    if (ERR_IS_ERR(ebufp))
33    {
34        PIN_ERR_LOG_EBUF(ERR_LEVEL_ERROR, "mta_export_fiscal_item:
              error preparing cod produto", ebufp);
35    }
36
37    strcat(file_line, cod_status);
38    strcat(file_line, cod_unid_medida);
39    strcat(file_line, quantidade);
40    vlr_unitario_dec = FLIST_FLD_GET(srch_res_flistp, VALOR, 1, ebufp
          );
```

Algoritmo 17 – Função de Geração de Arquivo de Item - Parte 3

```
    if (vlr_unitario_dec == NULL)
    {
        strcat(vlr_unitario, "000000000000000");
        strcat(file_line, vlr_unitario);
    }
    else
    {
        strcpy(temp_buffer, "\0");
        strcpy(temp_format, "\0");
        vlr_unitario = pbo_decimal_to_str(vlr_unitario_dec, ebufp);
        mta_fiscal_char_filler(vlr_unitario, 15, 0, temp_format);
        sprintf(temp_buffer, "%-15.15s", temp_format);
        strcat(file_line, temp_buffer);
    }
    if (ERR_IS_ERR(ebufp))
    {
        ERR_LOG_EBUF(ERR_LEVEL_ERROR, "mta_export_fiscal_item: error
           preparing valor unitario", ebufp);
    }

    vlr_bruto_dec = FLIST_FLD_GET(srch_res_flistp, VALOR_TOTAL, 1,
       ebufp);
    if (vlr_bruto_dec == NULL)
    {
        strcat(vlr_bruto, "000000000000000");
        strcat(file_line, vlr_bruto);
    }
    else
    {
        strcpy(temp_buffer, "\0");
        strcpy(temp_format, "\0");
        vlr_bruto = pbo_decimal_to_str(vlr_bruto_dec, ebufp);
        mta_fiscal_char_filler(vlr_bruto, 15, 0, temp_format);
        sprintf(temp_buffer, "%-15.15s", temp_format);
        strcat(file_line, temp_buffer);
    }
    if (ERR_IS_ERR(ebufp))
    {
        ERR_LOG_EBUF(PIN_ERR_LEVEL_ERROR, "mta_export_fiscal_item:
           error preparing valor bruto", ebufp);
    }

    strcat(file_line, lote);
    data_emissao = (char *)FLIST_FLD_GET(srch_res_flistp,
       DATA_EMISSAO, 1, ebufp);
    strcpy(temp_buffer, "\0");
    strcpy(temp_format, "\0");
    mta_fiscal_char_filler(data_emissao, 8, 0, temp_format);
    sprintf(temp_buffer, "%-8.8s", temp_format);
    strcat(file_line, temp_buffer);
```

Algoritmo 18 – Função de Geração de Arquivo de Item - Parte 4

```
1     if (ERR_IS_ERR(ebufp))
2     {
3         ERR_LOG_EBUF(ERR_LEVEL_ERROR, "mta_export_fiscal_item: error
              preparing dt_emissao", ebufp);
4     }
5
6     strcat(file_line, descricao_nota);
7     strcat(file_line, cod_grp_produto);
8     strcat(file_line, transacao);
9     strcat(file_line, operacao);
10    strcat(file_line, cliente);
11
12    cod_municipio = (char *)FLIST_FLD_GET(srch_res_flistp,
          FLD_MUNICIPIO, 1, ebufp);
13    strcpy(temp_buffer, "\0");
14    strcpy(temp_format, "\0");
15    pin_mta_ac_fiscal_space_filler(temp_format, cod_municipio, 10);
16    sprintf(temp_buffer, "%-10.10s", temp_format);
17    strcat(file_line, temp_buffer);
18    if (ERR_IS_ERR(ebufp))
19    {
20        ERR_LOG_EBUF(ERR_LEVEL_ERROR, "mta_export_fiscal_item: error
              preparing id item", ebufp);
21    }
22
23    strcat(file_line, dcr);
24    strcat(file_line, hash_code);
25    strcat(file_line, categoria_item);
26    strcat(file_line, qtde_contratada);
27    strcat(file_line, cod_finaliza);
28
29    //Enter
30    strcat(file_line, "\n");
31    if (ERR_IS_ERR(ebufp))
32    {
33        ERR_LOG_EBUF(ERR_LEVEL_ERROR, "mta_export_fiscal_item: Erro
              ao criar linha para adicionar no arquivo", ebufp);
34    }
35    //O arquivo precisa existir
36    if (fPtr != NULL)
37    {
38        pthread_mutex_lock(&lock);
39        if (lines >= MAX_LINES)
40        {
41            PIN_ERR_LOG_MSG(ERR_LEVEL_DEBUG, "mta_export_fiscal_item:
                  Numero de linhas ultrapassado. Criando novo arquivo."
                );
42            mta_fiscal_create_file(ctxp, op_in_flistp, ebufp);
43        }
```

Algoritmo 19 – Função de Geração de Arquivo de Item - Parte 5

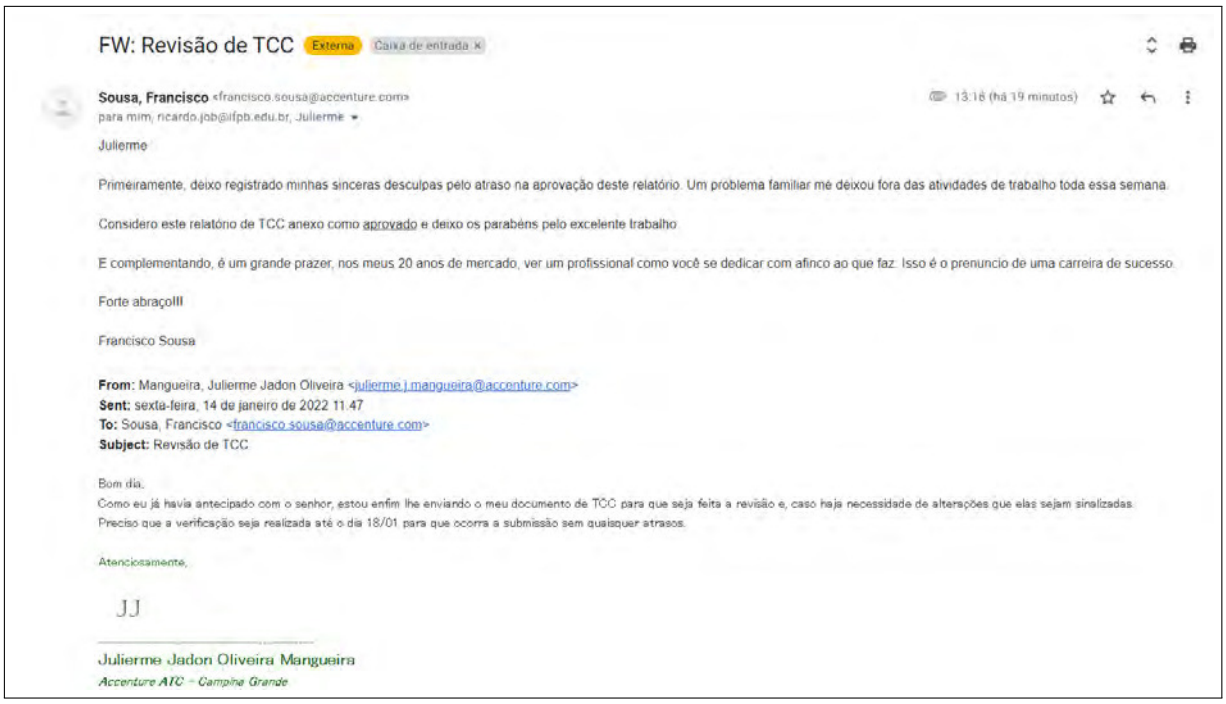
```
1          strcpy(filename, file_name);
2          lines++;
3          pthread_mutex_unlock(&lock);
4          ERR_LOG_MSG(ERR_LEVEL_DEBUG, "mta_export_fiscal_item: Nome do
               arquivo a escrever:");
5          ERR_LOG_MSG(ERR_LEVEL_DEBUG, filename);
6
7          // Abrindo novo arquivo
8          sprintf(file_name_final, "%s%s", FILE_DIRECTORY, filename);
9          fPtr = fopen(file_name_final, "a");
10         res = fprintf(fPtr, file_line);
11         if (res <= 0)
12         {
13             set_err(ebufp, ERRLOC_APP, ERRCLASS_SYSTEM_DETERMINATE,
                   ERR_PERMISSION_DENIED, 0, 0, 0);
14             ERR_LOG_EBUF(PIN_ERR_LEVEL_ERROR, "mta_export_fiscal_item
                   : error while creating and writing on file", ebufp);
15             goto cleanup;
16         }
17         // Chamada a outra funcao que atualiza o arquivo
18         mta_export_fiscal_update_filename(ctxp, srch_res_flistp,
               filename, ebufp);
19     }
20     // Liberar a memoria
21     cleanup:
22     return res;
23 }
```

Fonte: Elaborado pelo autor

APÊNDICE C – VALIDAÇÃO DE ANUÊNCIA DOS DADOS

Na figura 19 está demonstrado o e-mail recebido de um dos representantes legais da Accenture Campina Grande, como comprovação de anuência dos dados e desenvolvimentos aqui expostos neste trabalho.

Figura 19 – E-mail recebido como comprovação de anuência



FW: Revisão de TCC Externa Caixa de entrada ×

Sousa, Francisco <francisco.sousa@accenture.com> 13:18 (há 19 minutos)
para mim, ricardo.job@ifpb.edu.br, Julierme

Julierme

Primeiramente, deixo registrado minhas sinceras desculpas pelo atraso na aprovação deste relatório. Um problema familiar me deixou fora das atividades de trabalho toda essa semana.

Considero este relatório de TCC anexo como aprovado e deixo os parabéns pelo excelente trabalho.

E complementando, é um grande prazer, nos meus 20 anos de mercado, ver um profissional como você se dedicar com afinco ao que faz. Isso é o prenuncio de uma carreira de sucesso.

Forte abraço!!!

Francisco Sousa

From: Mangueira, Julierme Jadon Oliveira <julierme.j.mangueira@accenture.com>
Sent: sexta-feira, 14 de janeiro de 2022 11:47
To: Sousa, Francisco <francisco.sousa@accenture.com>
Subject: Revisão de TCC

Bom dia,
Como eu já havia antecipado com o senhor, estou enfim lhe enviando o meu documento de TCC para que seja feita a revisão e, caso haja necessidade de alterações que elas sejam sinalizadas.
Preciso que a verificação seja realizada até o dia 18/01 para que ocorra a submissão sem quaisquer atrasos.

Atenciosamente,

JJ

Julierme Jadon Oliveira Mangueira
Accenture ATC - Campina Grande

Fonte: Gmail, 2021.

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)

Trabalho de Conclusão de Curso

Assunto:	Trabalho de Conclusão de Curso
Assinado por:	Julierme Jadon
Tipo do Documento:	Projeto
Situação:	Finalizado
Nível de Acesso:	Ostensivo (Público)
Tipo do Conferência:	Cópia Simples

Documento assinado eletronicamente por:

- **Julierme Jadon Oliveira Mangueira**, **ALUNO (201412010233) DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - CAJAZEIRAS**, em 07/03/2022 09:58:22.

Este documento foi armazenado no SUAP em 07/03/2022. 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: 454168
Código de Autenticação: 97409b9f22