UNIVERSIDADE DO VALE DO RIO DOS SINOS – UNISINOS
UNIDADE ACADÊMICA DE EDUCAÇÃO ONLINE
ESPECIALIZAÇÃO EM ENGENHARIA DE SOFTWARE

Airton Fitarelli Junior

ESTUDO INVESTIGATIVO PRELIMINAR SOBRE ARTEFATOS GERADOS E/OU USADOS NO PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

Porto Alegre
2019

UNIVERSIDADE DO VALE DO RIO DOS SINOS – UNISINOS
UNIDADE ACADÊMICA DE EDUCAÇÃO ONLINE
ESPECIALIZAÇÃO EM ENGENHARIA DE SOFTWARE

Airton Fitarelli Junior

ESTUDO INVESTIGATIVO PRELIMINAR SOBRE ARTEFATOS GERADOS E/OU USADOS NO PROCESSO DE DESENVOLVIMENTO DE SOFTWARE

Trabalho de Conclusão de Curso apresentado como requisito parcial para a obtenção do título de Especialista em Engenharia de Software, pelo curso de Pós-Graduação Lato Sensu em Engenharia de Software da Universidade do Vale do Rio dos Sinos UNISINOS.
Orientador: Prof. MSc Guilherme S. Lacerda

Porto Alegre
2019

Estudo investigativo preliminar sobre artefatos gerados e/ou usados no processo de desenvolvimento de software

Airton Fitarelli Junior

Unidade Acadêmica de Pesquisa e Pós-Graduação – Universidade do Vale do Rio dos Sinos (UNISINOS)

Caixa Postal 90.470-280 – Porto Alegre – RS – Brasil

airtonfjunior88@gmail.com

Abstract. *The use of artifacts is very common in software development methods. However, with agile methodologies, they started to be less used, mostly because one of their premises is the use of less documentation. This study aims to identify, through a survey with 52 people, which are the artifacts more used during the process of software development, as well as their importance to the process and the consequences of their absence.*

Resumo. *O uso de artefatos vem sendo bastante utilizado nos métodos de desenvolvimento de software. Porém com advento das metodologias ágeis, seu uso começou a ser menos utilizado, sobretudo que uma das suas premissas é o uso de menos documentação. Este estudo busca identificar, através de uma pesquisa survey com 52 respondentes quais os artefatos mais usados durante o processo de desenvolvimento de software, bem como sua importância para o processo e suas consequências na sua falta.*

1. Introdução

A engenharia de software pode ser considerada uma ciência relativamente recente. A sua primeira aparição do termo acontece apenas em 1968, na Alemanha, na conferência de Engenharia de Software da OTAM [Wazlawick, 2013]. Nesse período eram realizadas entregas do produto apenas no final do processo de desenvolvimento, sendo esse modelo denominado métodos cascata. Outros métodos surgiram depois desse, como o incremental, RUP, entre outros.

Entretanto, todos esses métodos tinham como características gerar uma grande quantidade de documentos e burocratizar o processo de desenvolvimento de software. Os métodos ágeis surgiram como uma insatisfação das pesadas abordagens durante o período de desenvolvimento de software. Buscou-se com os métodos ágeis, entregas mais rápidas e diminuir a burocracia no desenvolvimento de aplicações de software [Sommerville, 2011].

Um dos itens do manifesto ágil é bem focado no que se diz respeito a valorização do software do que a documentação: “Software em funcionamento mais que documentação abrangente”. Nesse cenário, diversas empresas vêm utilizando metodologias ágeis para seu desenvolvimento de software e esse número vem crescendo nos últimos anos, desde o manifesto em 2001.

Entretanto, a proliferação dos métodos ágeis provocou um entendimento que a documentação em software é pouco necessária e seu uso deve ser suprimida em relação as entregas de software. Nesse ponto, criou-se um problema no desenvolvimento de software, uma vez que, os programadores recebem muito pouco ou nada de artefatos e isso podem ocasionar dificuldades no entendimento da solução e da necessidade do cliente.

Para isso, faz-se necessário encontrar um equilíbrio entre artefatos uteis para o desenvolvimento e manter a agilidade no desenvolvimento de software. Além identificar quais são os artefatos que geram valor para os times ágeis. Para tanto, o tema principal desse projeto pesquisa é: Quais são os artefatos ágeis que possam agregar valor no desenvolvimento de software e que sejam relevantes para o processo de construção de software?

Dessa forma, o presente estudo tem como objetivo principal identificar quais os principais artefatos gerados durante o processo de desenvolvimento de software, sendo esses a análise, design, desenvolvimento e testes. Como objetivos secundários, será levantado a importância do uso de artefatos no desenvolvimento de software, bem como o seu grau de importância para a construção do software.

Esse trabalho será organizado na seguinte maneira: Na seção 2 será apresentado o levantamento teórico desse artigo e na seção 3, será explanada alguns trabalhos relacionados. Já na seção 4, será explorado a metodologia utilizada para a pesquisa. Na seção 5 será apresentado os resultados obtidos e por fim, na última seção, a 6, será finalizado com as considerações finais.

2. Revisão de literatura

Nessa seção será apresentada a revisão da literatura sobre os modelos de desenvolvimento de software tradicional e as metodologias ágeis. Além disso, será explanado as etapas do desenvolvimento de software bem como os principais artefatos utilizados e/ou gerados nesse processo.

2.1 Modelos de desenvolvimento tradicional

Nessa seção serão apresentados alguns dos mais representativos modelos tradicionais de desenvolvimento de software reportados da literatura.

2.1.1 Modelo cascata

Quando se fala em desenvolvimento de software, o paradigma mais antigo se tem notícia é o conhecido como modelo cascata [Pressman, 2016]. Nesse modelo de desenvolvimento de software, o processo é realizado em etapas, isso é, ao iniciar uma etapa, todo esforço se concentra nessa ação e a próxima se inicia apenas quando a seguinte terminar. Esse modelo é caracterizado por planejar todas as etapas antes de iniciar o processo de desenvolvimento de software e entregar a solução ao cliente no final do processo [Sommerville, 2011].

As principais etapas do modelo de desenvolvimento de software em cascata podem ser descritas como [Sommerville, 2011]:

- Análise e definição de requisitos: Onde são levantadas necessidades e especificações do sistema a ser desenvolvido.
- Projeto de sistema de software: Busca identificar e descrever as funcionalidades do sistema e seus relacionamentos.
- Implementação e teste unitário: É realizado a implementação das funcionalidades apresentadas, bem como os testes que garantem seu funcionamento.
- Integração e teste de sistema: Busca-se realizar os testes por completo e verificar que os requisitos foram atingidos. Caso tudo tenha êxito, o sistema é entregue para o cliente.
- Operação e manutenção: Nessa última fase são realizados a correção de erros reportados pelo cliente e possíveis novas necessidades não identificadas na fase de análise.

Sua utilização deve ser aplicada em modelos de software onde os requisitos são claramente conhecidos e com pouca possibilidade de mudança durante o ciclo de desenvolvimento [Sommerville, 2011].

Esse modelo de desenvolvimento de software tem como característica que as mudanças e possíveis falhas na etapa de análise e definição sejam apenas perceptível no final do processo. Por isso, esse modelo não é recomendado nos dias atuais, sobretudo pelas mudanças constantes do processo de desenvolvimento de software [Pressman, 2016].

2.1.2 Modelo Incremental

O modelo incremental de desenvolvimento de software tem como característica realizar pequenos incrementos ao invés de entregar toda a solução de uma só vez. Com isso, é possível realizar a solução em diversos entregáveis, possibilitando a validação e feedback entre as entregas [Sommerville, 2011].

Suas principais vantagens, sobretudo com o modelo cascata são:

- As mudanças do projeto são melhor acomodadas, uma vez que, a cada entrega o cliente apresenta seu feedback da solução;
- Maior facilidade de obtenção de feedback do cliente devido das entregas incrementais.

Porém a abordagem incremental tem algumas desvantagens sob o ponto de vista do gerenciamento [Sommerville, 2011].

2.1.3 RUP

O método de desenvolvimento de software conhecido como RUP, de *Rational Unified Process*, surge na necessidade de criar um processo que seja incremental, iterativo e baseado em casos de uso. Esse modelo busca utilizar as melhores características dos modelos tradicionais e mas com constante participação do cliente e entregas incrementais e iterativo [Pressman, 2016].

Segundo Sommerville (2011), o processo RUP é dividido em 4 fases:

- Concepção: Nessa fase, será identifica as necessidades do cliente e o suas regras de negócio.
- Elaboração: Na elaboração será desenvolvido a compreensão do problema reportado pelo cliente. Ao final serão gerados dos casos de uso da UML e um plano de desenvolvimento de software.
- Construção: Nessa fase é desenvolvido o software e realizado, os testes de sistema e as documentações necessárias. Ao final dessa etapa, já teremos o software em funcionamento.
- Transição: Nessa etapa é liberado ao cliente o software desenvolvido bem como sua documentação de funcionamento.

Além disso, Sommerville (2011), explica que "A visão estática do RUP prioriza as atividades que ocorrem durante o processo de desenvolvimento. Na descrição do RUP, essas são chamadas workflows". Esses workflows podem ser os seguintes:

- Modelagem de negócio: Os processos de negócio são modelados pro meio de casos de uso de negócios
- Requisitos: Atores que interagem com o sistema são identificados e casos de uso são desenvolvidos para modelar os requisitos do sistema
- Análise e projeto: Um modelo de projeto é criado e documentado com modelos de arquitetura, modelos de componentes, modelos de objetos e modelos de sequencia
- Implementação: Os componentes do sistema são implementados e estruturados em subsistemas de implementação. A geração automática de código a partir de modelos de projeto ajuda a acelerar esse processo
- Teste: O teste é um processo iterativo que é feito em conjunto com a implementação. O teste de sistema segue a conclusão da implementação
- Implantação: Um release do produto é criado, distribuído aos usuários e instalado em seu local de trabalho.
- Gerenciamento de configuração e mudanças: Esse workflow de apoio gerencia as mudanças do sistema
- Gerenciamento de projetos: Esse workflow de apoio gerencia o desenvolvimento do sistema
- Meio ambiente: Esse workflow está relacionado com a disponibilização de ferramentas apropriadas para a equipe de desenvolvimento de software

2.2 Metodologias ágeis

Os métodos ágeis surgiram na década de 1990 como uma resposta a insatisfação do modelo atual a época, que consistia em focar primariamente no planejamento e na análise das funcionalidades ao invés do desenvolvimento do software.

Segundo Ambler (2018), a definição de modelagem ágil pode ser "uma coleção de valores, princípios e práticas para software de modelagem que pode ser aplicado em

um projeto de desenvolvimento de software de uma maneira eficaz e leve". O manifesto ágil foi construído sobre quatro pilares fundamentais, sendo esses descritos no manifesto ágil:

- Indivíduos e interações mais do que processos e ferramentas;
- Software funcional mais do que documentação abrangente;
- Colaboração com o cliente mais do que negociação contratual;
- Responder à mudança mais do que seguir um plano;

Dentre as diversas metodologias ágeis, será buscado conceituar algumas delas

2.2.1 Extreme Programming (XP)

A metodologia ágil chamada de Extreme Programming, ou XP, é talvez a mais conhecida e utilizada [Sommerville, 2011]. A mesma possui como principais características o feedback constante, abordagem incremental e encoraja a comunicação entre as partes envolvidas [Sbrocco & Macedo, 2012].

Além disso podemos destacar algumas características do XP como a programação em pares, onde os programadores trabalham juntos na mesma estação de trabalho e com isso buscam desenvolver códigos melhores, continua revisão e refatoração de código constante [Sommerville, 2011].

Outra característica importante do XP é o feedback constante, uma vez que, o método estimula o contato do desenvolvedor com o cliente. A partir das entregas realizadas existe uma maior interação entre as a partes e com isso melhor entendimento da solução bem como de possíveis erros do sistema [Sbrocco & Macedo, 2012].

2.2.2 Scrum

A metodologia ágil Scrum é outra abordagem ágil, que tem no seu nome uma jogada de rugby, onde os jogadores se reúnem para reiniciar o jogo. [Sbrocco & Macedo, 2012]. Diferente do XP, o Scrum é mais focado do gerenciamento do projeto e não em práticas de desenvolvimento.

O Scrum, segundo Sommerville (2011), pode ser dividida em três etapas

- Planejamento: Onde se estabelece os objetivos do projeto e a arquitetura que será abordada;
- Ciclos de Sprints: Onde em cada sprint ocorre um incremento de software ao sistema;
- Encerramento: Onde é desenvolvido os manuais e uso e toda a retrospectiva e lições aprendidas do processo.

2.3 Etapas do desenvolvimento de software

Os artefatos coletados na literatura foram os listados na tabela 1 e identificados nas seguintes etapas.

Tabela 1 – Artefatos e sua etapa de uso

Artefato	Etapa de análise	Etapa de Design	Etapa de Desenvolvimento	Etapa de testes
Casos de teste				x
Classe de colaboração de responsabilidade (CRC)	x			
Diagrama de atividades	x	x	x	
Diagrama de classes	x	x	x	
Diagrama de casos de uso	x	x		x
Diagrama de componentes			x	
Diagrama de estados			x	
Diagrama de objeto			x	
Diagrama de robustez	x			
Diagrama de sequência			x	
Descrição de caso de uso	x	x		
Documento de requisitos	x	x	x	x
Documento de requisições técnicas	x			
Documento de restrições do sistema	x			
Glossário	x			
Regras de negócio	x			
Mapa mental	x			
Mapeamento	x			

objeto-relacional (ORM)				
Modelo lógico de dados (LDM)	x			
Product Backlog			x	x
Protótipo de interface de usuário		x	x	
Regras de negócio		x		
TaskBoard			x	
Testes de aceitação			x	
Users Stories	x	x		x
UI StoryBoards		x		

2.3.1 Análise

Essa etapa corresponde o processo inicial no desenvolvimento de software, onde será coleta os requisitos do sistema junto ao cliente. Segundo Pressman (2011) “A identificação da necessidade é o ponto de partida na evolução de um sistema baseado em computador”.

Após os requisitos coletados é realizado toda a modelagem do sistema. Sommerville (2011) descreve a modelagem de sistema como “o processo de desenvolvimento de modelos abstratos de um sistema, em que cada modelo apresenta uma visão ou perspectiva diferente do sistema”. Essa modelagem pode ser através de notações gráficas, como a UML, do inglês *Unified Modeling Language*.

2.3.2 Design

A etapa de design busca compreender como o sistema deve ser organizado e a sua estrutura geral. Nessa etapa será planejada toda a arquitetura da solução bem como eles se como os componentes se comunicam [Sommerville, 2011].

Segundo Pressman (2011), “O projeto de software encontra-se no núcleo técnico do processo de engenharia de software e é aplicado independentemente do paradigma de desenvolvimento usado”.

2.3.3 Desenvolvimento

A etapa de desenvolvimento é o momento onde se implementa todo projeto desenvolvido. Segundo Sommerville (2011), “é um estágio do processo no qual um sistema de software executável é desenvolvido”.

Nesse momento os programadores vão transformar todo as necessidades dos clientes em produtos entregáveis. Caso as demandas do cliente não forem claramente entendido pelos desenvolvedores ocorre grande risco do produto realizado não atender as necessidades do cliente.

2.3.4 Testes

A etapa de testes é responsável por validar o produto criado pela etapa de desenvolvimento. Se buscará validar se o sistema está funcionando sem erros e se está condizente com as regras de negócio do sistema. Sua importância é devida que a construção do software é complexa e diversos erros podem acontecer durante o processo de desenvolvimento. [Jino at all, 2000].

Segundo Sommerville (2011), o processo de testes tem dois objetivos distintos:

- Demonstrar ao cliente que o sistema atende as necessidades solicitadas. Isso pode ser através da validação dos requisitos previamente levantados.
- Identificar situações onde o sistema apresenta comportamento indesejável, incorreto ou diferente das especificações propostas.

3. Trabalhos Relacionados

Nessa seção são abordados alguns trabalhos relacionados com o tema proposto. O foco das pesquisas foi em artigos relacionados a artefatos e o desenvolvimento de software.

Um estudo realizado na Holanda em 2011 já buscava identificar a importância dos artefatos nas metodologias ágeis e como os profissionais de software os enxergam. Nesse estudo foi realizado um *survey* para identificar a importância dos artefatos de software. Os resultados revelaram que os entrevistados entendem a importância dos artefatos no processo de desenvolvimento de software porem o consideram um “fardo” no processo. Foi percebido também que a documentação ágil precisa ser melhor adaptada para agregar valor [Stettina & Heijstek, 2011].

Outro estudo, de Wagenaar at all (2015) sobre o uso de artefatos no desenvolvimento de software foi um artigo que buscou identificar os artefatos usados nas metodologias ágeis. Porem o mesmo estudo buscou identificar que mesmo nas metodologias ágeis, quais outros artefatos também eram utilizados. O resultado encontrou que além dos artefatos ágeis, os times usam outros tipos de artefatos, nas etapas de projeto, teste e liberação do produto.

O mesmo autor escrever outro artigo buscando identificar as razões para quais as equipes ágeis utilizam artefatos. Alguns aspectos obtidos foram no estudo foram, segundo os autores: O uso de metodologias ágeis leva ao uso de artefatos ágeis, como product backlog, epic, etc. Comunicação interna, garantia da qualidade através de artefatos de teste, controle de atividades.

4. Metodologia de pesquisa

Nessa seção do trabalho será apresentado a metodologia de trabalho. Será explanado desse seu planejamento até sua execução.

A primeira etapa desenvolvida foi a revisão da literatura, através de livros, periódicos, artigos e demais fontes que busquem identificar aos diversos tipos de artefatos propostos para desenvolvimento de software. A busca dessas informações coletadas por outros autores permite contribuir com a pesquisa a ser desenvolvida, quer que seja para demostrar contradições ou reafirmar comportamentos [Marconi, 2017].

Para o presente trabalho foram coletadas informações sobre as diferentes formas de processo de desenvolvimento de software tradicional. Além disso, será abordado também as metodologias ágeis e suas diferenças comparado aos métodos tradicionais. Outro ponto abordado foram as diferentes etapas no processo de software, sendo eles divididos em quatro etapas: análise, design, desenvolvimento e testes. Além disso foram analisados os diversos artefatos que são gerados e/ou usados durante o processo.

Após coletar as informações da literatura o presente trabalho buscou realizar um survey. Uma vez que será buscado informações de um grupo especifico de indivíduos [Gerhardt & Silveira, 2009]. Já a realização do questionário será dará através um formulário eletrônico disponibilizado principalmente nas redes sociais. Acredita-se que essa forma, isto é, as redes sociais sejam possíveis conseguir uma maior capacidade de penetração e respostas obtidas.

Segundo Marconi (2017) o "questionário é um instrumento de coleta de dados, constituído por uma série ordenada de perguntas, que devem ser respondidas por escrito e sem a presença do entrevistador". Além disso, o questionário é a principal fonte de obtenção de dados. [Barros & Lehfeld, 2007].

Para elaboração desse questionário, foi dividido nos seguintes aspectos:

- Perfil do Respondente: Buscou identificar o perfil do respondente, baseado na sua profissão, suas experiencias profissionais, tempo de experiencia e experiência em métodos ágeis;
- Artefatos na etapa de análise de software: O objetivo foi identificar como são buscadas as informações para levantar requisitos, os artefatos gerados e sua importância;
- Artefatos na etapa de design do software: Seu objetivo foi coletar informações sobre os artefatos mais utilizados pelos desenvolvedores, sua importância e os impactos na sua falta;
- Artefatos na etapa de desenvolvimento: O objetivo foi mapear as características desse processo sobretudo sobre os documentos gerados nesse processo, o impacto da falta de especificação e sua importância.
- Artefatos na etapa de teste: Buscou-se identificar quais artefatos geram mais valor no processo de teste, sua importância e a sua falta de especificação.

Na construção do questionário, utilizou-se diversos tipos de perguntas. Para buscar informações sobre os artefatos mais utilizados foi utilizado perguntas de múltipla escolha com alternativas previamente elaborados com base no referencial teórico. Além disso foi disponibilizado um campo livre para o usuário preencher outra opção.

Outro tipo de pergunta apresentada para buscar a importância de diversos temas. Para tanto foi utilizado a Escala Likert para obter esse tipo de informação. A escala de

Likert é composta por frases que buscam buscar o grau de condordância entre discordo totalmente até o concordo totalmente, em níveis de 5 7 ou 11. [Cunha, 2007] A Além disso, a escala Likert foi utilizada para identificar a importâncias dos artefatos das diversas fases de desenvolvimento de software.

Após o questionário produzido, esse foi submetido para um primeiro teste onde foi submetido para três pessoas. Essa foram consideradas “testes” do mesmo. O perfil utilizado para essas três pessoas foram: um professor universitário, um analista de sistemas com 20 anos de experiência e um programador do nível Junior. Após as considerações dos mesmos, foi realizada as correções sugeridas e preparado para envio do grande público. As ponderações foram em deixar mais claras as perguntas abertas e melhorar o texto das mesmas.

Para envio do questionário, foi utilizado o formulário do Google como forma de obtenção das respostas. A sua escolha foi devida o mesmo ser uma ferramenta open source e fácil entendimento das pessoas que responderão o mesmo. Para divulgação, foi utilizado as redes sociais, sobretudo Facebook e Linkedin. Na rede social Facebook foi disponibilizado através do perfil do autor desse artigo além de grupos relacionados a TI e desenvolvimento de software. Também foi divulgado no perfil social do autor da rede social Linkedin.

O objetivo do questionário é buscar pessoas que tenham experiência em desenvolvimento de software e tenham conhecimento e/ou utilizem artefatos durante o desenvolvimento de software. O questionário foi disponibilizado no dia 5 de maio de 2019 e foi encerrado a coloca de informações no dia 23 de junho, tendo um período de coleta de informações de 49 dias. Após o período de coleta de informações, o questionário alcançou o número de 52 respondentes.

Com as informações coletas iniciou-se a análise dos resultados. Para buscar organizar as informações, utilizou-se a mesma ferramenta do Google da realização do questionário. Com ela foi possível analisar exportar os resultados e identificar seu valor.

Os dados quantitativos foram organizados em tabelas e buscado o valor percentual de cada resposta. Entretanto, nas questões abertas, foi realizada a análise qualitativa dos dados. Nesse tipo de análise, é necessário examinar os dados, buscando identificar falhas, distorções ou erros. Com os dados examinados, é realizado a classificação, codificação e tabulação [Barros & Lehfeld, 2007].

O mesmo autor entende que após a tabulação dos dados, é necessário a interpretação dos mesmos para buscar significado nos resultados. Segundo Barros e Lehfeld (2007), “terminando a interpretação dos dados, passará à montagem do relatório fim da pesquisa”.

5. Apresentação dos Resultados da Pesquisa

Das 52 respostas obtidas, ambas foram analisadas para verificar se as mesmas continham alguma incoerência e todas foram validas como aptas para o presente estudo. Após essa etapa, os dados serão divididos em 5 grupos, sendo esse: Dados do perfil do respondente, artefatos no processo de análise de software, artefatos no processo de design de software, artefatos no processo de desenvolvimento de software e artefatos no processo de teste de software.

5.1 Perfil do respondente

Nessa etapa buscou identificar o perfil do respondente do questionário. Para tanto, o objetivo era identificar o cargo atual do respondente, tento de experiencia em desenvolvimento de software, tipos de trabalhos que participou e se o mesmo trabalha com métodos ágeis.

Em relação ao cargo dos respondentes mais da metade, 52,2 %, são desenvolvedores. Um pouco mais de um quarto, 26,1%, são analistas. Já o cargo de PO foi respondido por 8,7% e os cargos de gerente de desenvolvimento, Scrum master e gerente de projetos obteve cada um 4,3%. Não houve respondentes no cargo de testador.

No que se refere ao tempo de experiencia com desenvolvimento de software, a maioria dos respondentes, 39,1 %, foi com 10 a 20 anos de experiência. Já a segunda faixa de experiencia foi entre 5 a 10 anos, com 30,4 %. Com 13%, ficaram as faixas de experiencia entre 2 a 5 anos e acima de 20 anos. Já o tempo experiencia até 2 anos foi de 4,3%.

Já os tipos de trabalho na qual o profissional trabalhou obteve os seguintes resultados, uma vez que o respondente poderia escolher mais de uma opção. O resultado pode ser obtido na figura 1.

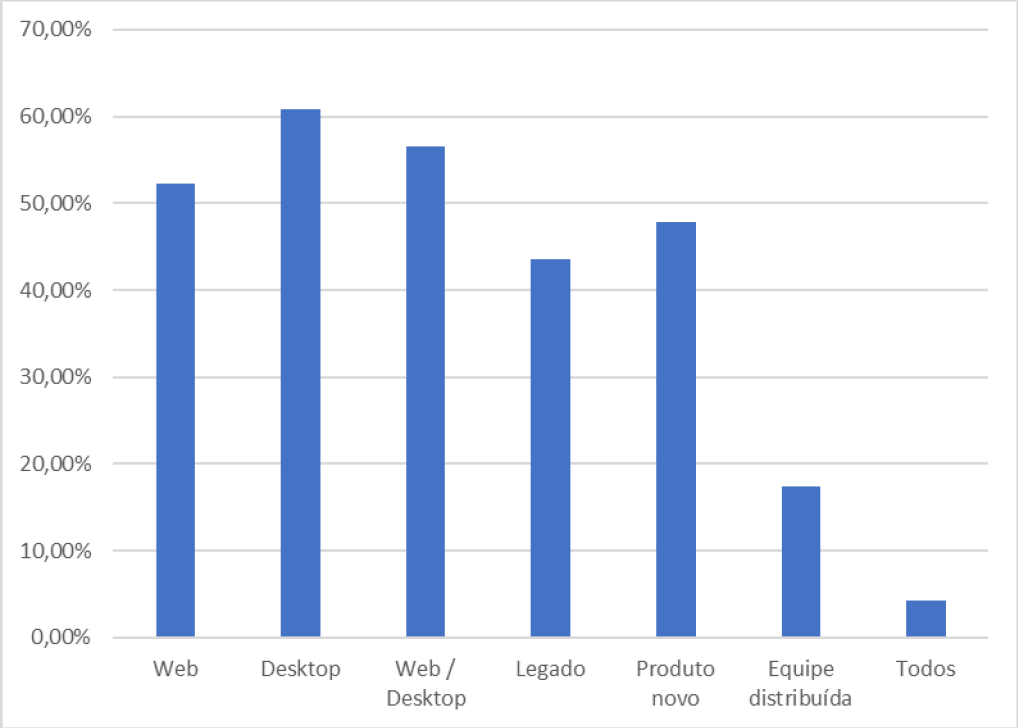

Figura 1 - Gráfico que representa em: Que tipo de projetos você já trabalhou?

Por fim foi questionado se o responde já trabalhou com métodos ágeis e a grande maioria respondeu que sim, tendo o percentual de 82,6 %. Já 17,4 % responderam que não.

5.2 Artefatos na análise de desenvolvimento de software

Nessa etapa foram abordados a foram de coletar requisitos e quais artefatos gerados/usados no processo de análise de software. Outro ponto abordado foi a importância dos mesmos e como eles podem afetar no processo.

A primeira questão referente a etapa de análise abordou quais técnicas são utilizadas para coletar requisitos. Nessa questão, o entrevistado poderia escolher mais de uma opção ou até mesmo informas outra que não estava previamente listada. A opção que teve mais votos foi as entrevistas não presencias, com 82,6%. Já a entrevista presencial também obteve uma grande votação, atingindo 73,9%. Em terceiro, a opção de observação obteve 65,2%. A opção pré-definida observação atingiu 17,4%. Nessa questão foram listadas outras opções dos respondentes como as seguintes: Experiencia de mercado, mercado e entrevista presencial e não presencial, sendo essas marcadas uma vez cada.

O segundo questionamento foi quais documentos usados no processo de analise de software. Os itens com * foram informados pelos respondentes e não estavam na lista de pré-definidos pelo autor. Os resultados foram listados na figura 2.

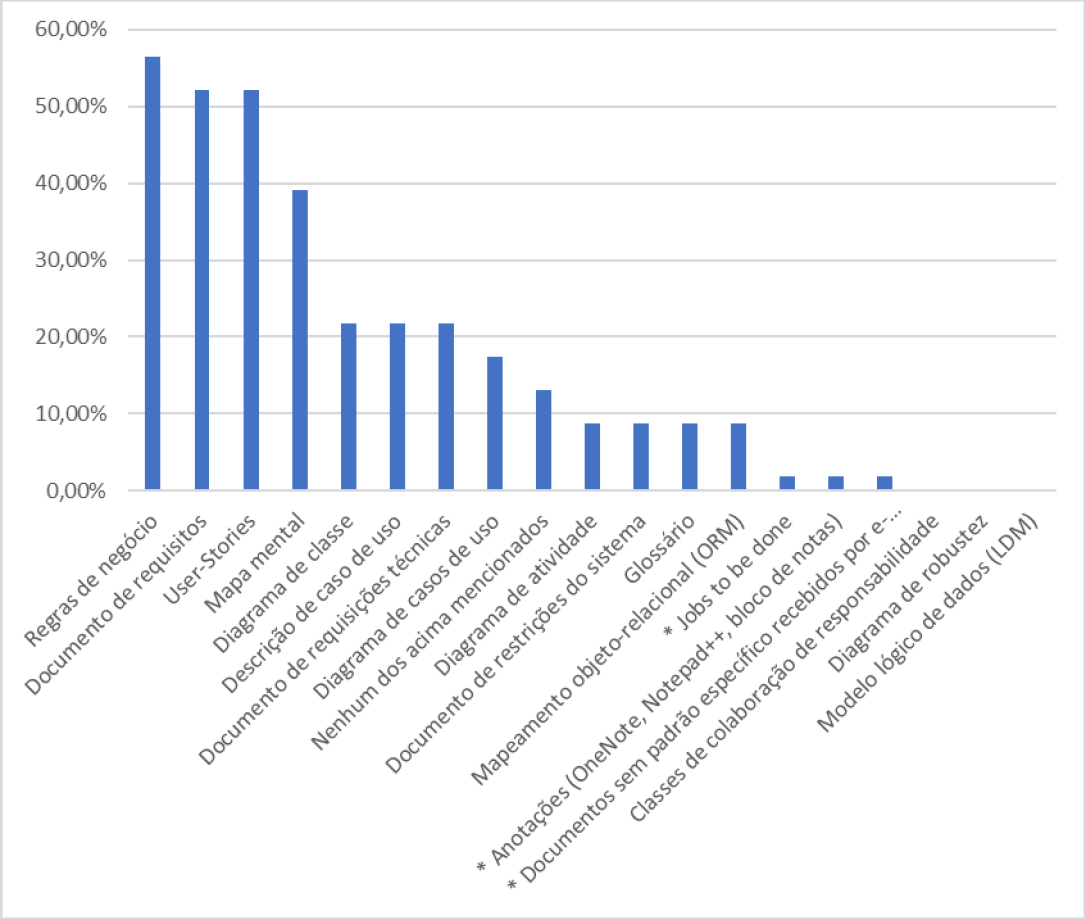

Figura 2 - Documentos usados na etapa de análise de software

Outro importante questionamento foi medir a importância dos documentos de analise no processo de software os resultados foram os abaixo citados, na tabela 2.

Tabela 2 - Documentos usados na etapa de análise de software e sua importância

	Muito importante	Importante	Neutro	Desnecessário	Muito desnecessário
Classes de colaboração de responsabilidade	0,0%	0,0%	60,9%	21,7%	17,4%
Descrição de caso de uso	17,4%	47,8%	30,4%	0,0%	4,3%
Diagrama de atividade	4,3%	30,4%	43,5%	13,0%	8,7%
Diagrama de casos de uso	26,1%	43,5%	21,7%	4,3%	4,3%
Diagrama de classe	13,0%	21,7%	39,1%	13,0%	13,0%
Diagrama de robustez	4,3%	4,3%	56,5%	21,7%	13,0%
Documento de requisições técnicas	17,4%	26,1%	43,5%	4,3%	8,7%
Documento de requisitos	26,1%	52,2%	13,0%	0,0%	8,7%
Documento de restrições do sistema	13,0%	34,8%	39,1%	4,3%	8,7%
Glossário	4,3%	13,0%	56,5%	13,0%	13,0%
Mapa mental	4,3%	26,1%	52,2%	8,7%	8,7%
Mapeamento objeto-relacional (ORM)	13,0%	17,4%	47,8%	13,0%	8,7%
Modelo lógico de dados (LDM)	8,7%	13,0%	60,9%	8,7%	8,7%
Regras de negócio	52,2%	34,8%	8,7%	4,3%	0,0%
User-Stories	30,4%	26,1%	39,1%	4,3%	0,0%

Outro ponto abordado foram os motivos das escolhas da pergunta anterior. Essa questão foi aberta foram listas as opções onde os respondentes escreveram um texto válido. As respostas obtidas foram as seguintes, conforme a tabela 3.

Tabela 3 – Respostas referente aos motivos das escolhas dos documentos na fase de análise

burocracia desnecessária
de acordo com conhecimento de aula com o tipo de informação que contém no contexto
utilidade para definição do produto/serviço
O caso de uso é o mais importante na minha área, pois é ali que se identifica a real necessidade do usuário, visto que o mesmo tem muitas dificuldades em utilizar/ou preencher documentos mais técnicos
Praticidade, clareza, ajuda na análise todos favorecem o foco no problema, o importante é entregar valor.
Software orientado a documentação não resolve problema do cliente
Foco na necessidade dos cliente x foco nos requisitos
Entendo que são de fácil compreensão e que qualquer pessoa envolvida no projeto tem condições de usar/colaborar.
Necessidade
Creio que uma junção dos conhecimentos acadêmicos e is padroes de trabalhos adotados ao longo dos anos
Escolhi esses documentos pois eles ajudam a identificar melhor a necessidade do cliente, como ele utiliza ou pretende utilizar o programa e quais limitações podem ocorrer ao desenvolver o programa.
Para se ter clareza no desenvolvimento do software.
No modelo agil aplicado hoje em dia o processo deve ser o menos burocrático possível, mais importante que dezenas de documentos é um levantamento de requisitos acertivo pra que a equipe possa realizar a grooming e planejar as tarefas de forma clara.
Os que deixei Neutro são os que não utilizo ou não tenho conhecimento. Os mais importantes são os que conheço e utilizo.
pelo uso no dia a dia
Por adotarmos atualmente uma metodologia ágil e evolutiva, baseada em User Stories e integrada ao ambiente desenvolvimento. Porém, entendo a importância de Use Cases e outros artefatos, para ambientes de desenvolvimento distribuídos.

Outra questão abordada foi se os documentos de análise são suficientes para o ambiente de trabalho. A opção com mais votos foi concordo parcialmente com 60,9%, Já a opção concordo totalmente obteve 17,4%. A opção indiferente ficou como a terceira mais votada com 13% e com 8,7% obteve a opção discordo totalmente. A opção discordo parcialmente não obteve votos.

A ultima questão abordada foi se os documentos são claros para o entendimento das necessidades do cliente. A opção concordo parcialmente obteve 56,5 % e foi a mais

votada. A opção concordo totalmente foi a segunda mais votada e obteve 26,1 %. A opção discordo totalmente e indiferente obtiveram 8,7%. A opção discordo totalmente não obteve votos.

5.3 Artefatos no design de desenvolvimento de software

Nessa etapa foram abordados quais artefatos gerados/usados no processo de design de software. Outro ponto abordado foi a importância dos mesmos e como eles podem afetar no processo.

A primeira questão abordada foi quais documentos são usados na etapa de design. Os resultados e os percentuais foram apresentados na figura 3.

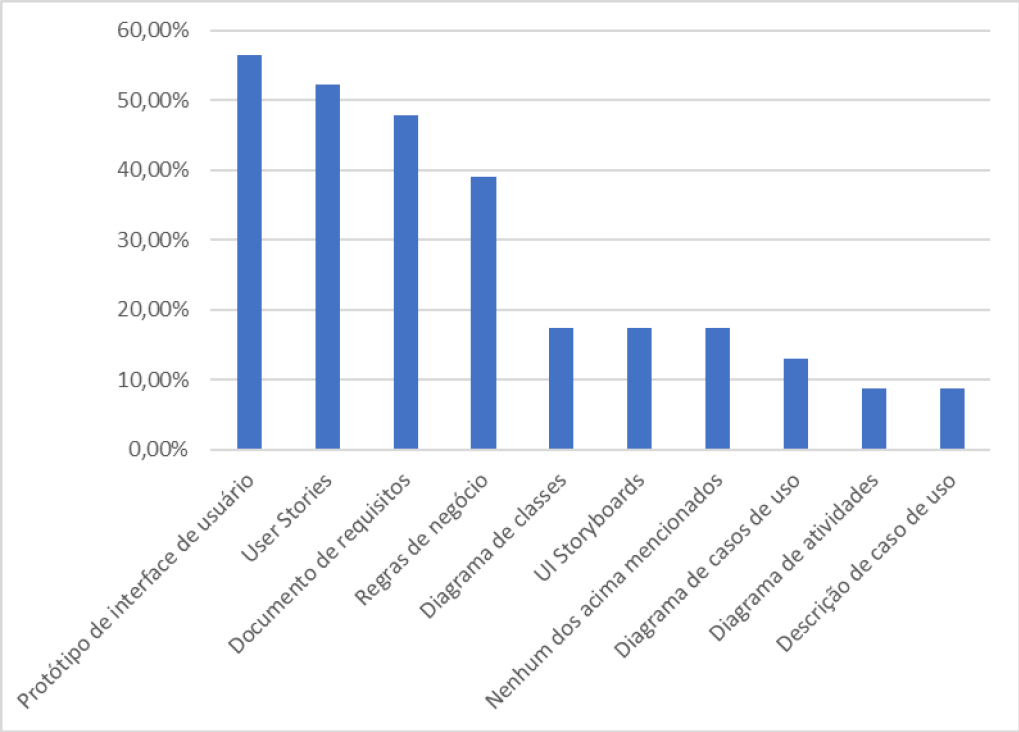

Figura 3 - Documentos usados no processo de design de software

Outro importante questionamento foi medir a importância dos documentos de design no processo de software os resultados foram os abaixo citados, na tabela 4.

Tabela 4 - Documentos usados na etapa de design de software e sua importância

	Muito importante	Importante	Neutro	Desnecessário	Muito desnecessário
Diagrama de atividade	4,3%	8,7%	69,6%	8,7%	8,7%
Diagrama de classe	0,0%	26,1%	56,5%	4,3%	13,0%
Diagrama de casos de uso	8,7%	26,1%	52,2%	0,0%	13,0%

Descrição de caso de uso	17,4%	26,1%	43,5%	0,0%	13,0%
Documento de requisitos	30,4%	26,1%	34,8%	0,0%	8,7%
Protótipo de interface de usuário	47,8%	26,1%	26,1%	0,0%	0,0%
Regras de negócio	43,5%	26,1%	26,1%	0,0%	4,3%
User-Stories	30,4%	21,7%	43,5%	0,0%	4,3%
UI Storyboards	4,3%	8,7%	69,6%	8,7%	8,7%

Outro ponto abordado foram os motivos das escolhas da pergunta anterior. Essa questão foi aberta foram listas as opções onde os respondentes escreveram um texto válido. As respostas obtidas foram as seguintes, conforme a tabela 5.

Tabela 5 – Respostas referente aos motivos das escolhas dos documentos na fase de design

Entendimento da tela ou fluxo de telas
Mesmo motivo da pergunta anterior
A mesma da anterior
Software funcionando mais do que documentação abrangente
Proporcionam um melhor detalhamento para o DEV
Acredito que trazem clareza ao projeto
Apresentar uma previa ao usuário
Lidar com design para mim é complexo. Tentar juntar o que o usuário vê com o que ele espera que seja e no final alinhar isto com a aplicação é complexo, acredito que por isto de minhas escolhas.
Acredito que com esses documentos é possível definir melhor o design do produto.
Atingir uma melhor "User Experience"
Assim como na resposta anterior, prioridade em ser objetivo e menos burocrático.
Não consigo argumentar, pois depois da faculdade, nunca trabalhei com documentação de projeto.
Mesma resposta que a anterior, os Neutros são os que não uso ou não conheço. Demais acredito que sejam importantes pois auxiliam num entendimento claro dos requisitos.
Para projetar uma boa interface é necessário entendimento das regras de negócio e o fluxo de trabalho.
Usablididade
As partes concordarem com o que tem que ser desenvolvido.

pelo uso
Porque o design tem uma grande dependência do conhecimento do que deve ser resolvido, bem como definir detalhadamente e minimamente os elementos de UI.
Facilidade de uso

Outra questão abordada foi se os documentos de design são suficientes para o ambiente de trabalho. A opção com mais votos foi concordo parcialmente com 52,2%, já a opção concordo totalmente e indiferente obtiveram cada uma 17,4 %. A opção discordo totalmente obteve 8,7% e menos votada foi a opção discordo parcialmente com 4,3%.

A última questão abordada foi se os documentos são claros para o entendimento das necessidades do cliente. A opção concordo parcialmente obteve 52,2 % e foi a mais votada. A opção concordo totalmente foi a segunda mais votada e obteve 30,4 %. A terceira mais votada foi indiferente, com 8,7% A opção discordo totalmente e indiferente obtiveram 4,3%.

5.4 Artefatos na etapa de desenvolvimento de software

Nessa etapa foram abordados quais artefatos gerados/usados no desenvolvimento de design de software. Outro ponto abordado foi a importância dos mesmos e como eles podem afetar no processo.

O primeiro questionamento foi quais documentos são usados no processo de desenvolvimento de software. Os valores e percentuais são apresentados na figura 4. O valor com asterisco foi informado pelo respondente.

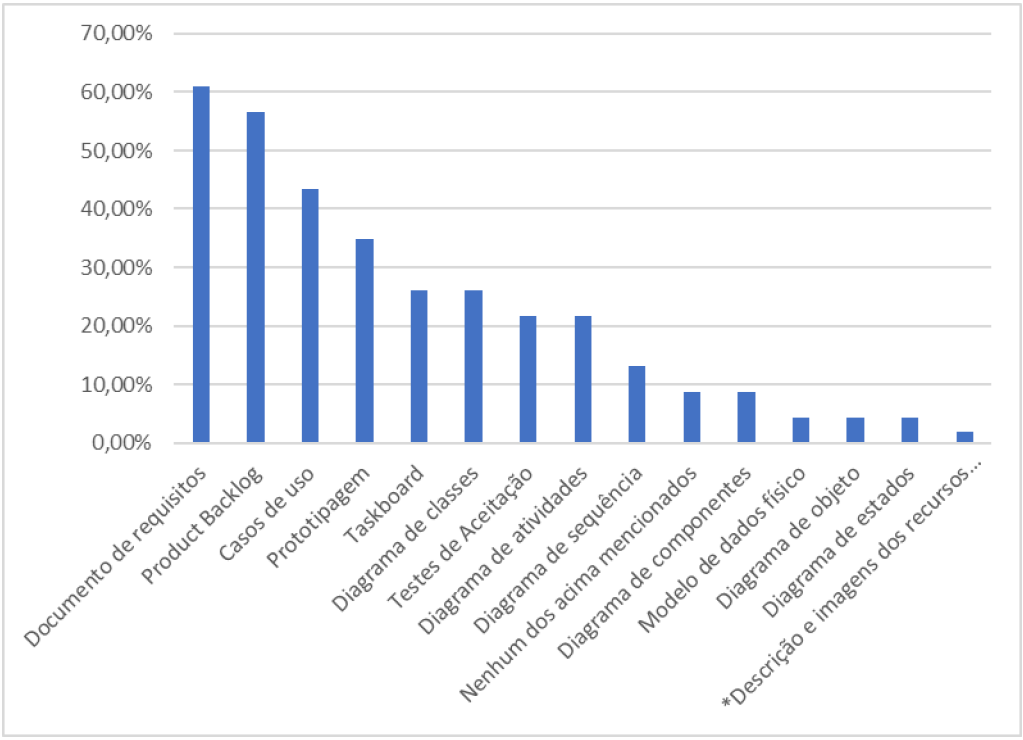

Figura 4 - Documentos usados na etapa de desenvolvimento de software

Outro importante questionamento foi medir a importância dos documentos de desenvolvimento no processo de software os resultados foram os abaixo citados, na tabela 6.

Tabela 6 - Documentos usados na etapa de desenvolvimento de software e sua importância

	Muito importante	Importante	Neutro	Desnecessário	Muito desnecessário
Casos de uso	8,7%	47,8%	30,4%	0,0%	13,0%
Diagrama de atividades	13,0%	21,7%	47,8%	4,3%	13,0%
Diagrama de classes	8,7%	26,1%	26,1%	8,7%	13,0%
Diagrama de componentes	4,3%	13,0%	56,5%	13,0%	13,0%
Diagrama de estados	0,0%	8,7%	69,6%	8,7%	13,0%
Diagrama de objeto	0,0%	4,3%	73,9%	4,3%	17,4%
Diagrama de sequência	4,3%	26,1%	56,5%	4,3%	8,7%
Documento de requisitos	26,1%	39,1%	21,7%	0,0%	13,0%
Modelo de dados físico	4,3%	13,0%	60,9%	0,0%	21,7%
Product Backlog	26,1%	39,1%	30,4%	4,3%	0,0%
Prototipagem	30,4%	43,5%	26,1%	0,0%	0,0%
Taskboard	8,7%	13,0%	60,9%	4,3%	13,0%
Testes de Aceitação	8,7%	34,8%	47,8%	4,3%	4,3%

Outro ponto abordado foram os motivos das escolhas da pergunta anterior. Essa questão foi aberta foram listas as opções onde os respondentes escreveram um texto válido. As respostas obtidas foram as seguintes, conforme a tabela 7.

Tabela 7 – Respostas referente aos motivos das escolhas dos documentos na fase de design

quanto menos melhor
O que utilizo no dia a dia
Mesmo motivo anterior

A mesma da anterior
Menos é mais
Muitos docmentos podem acabar atrapalhando o DEV
Acredito que com esses, os desenvolvedores já conseguem dar sequencia as atividades.
Sem comentários
Neutro
Acredito que são importantes para auxiliar o programador a desenvolver melhor e com mais rapidez um produto.
Agilidade no desenvolvimento.
De forma mais genérica geralmente durante o processo de desenvolvimento, precisamos do levantamento das atividades para documentação do siftware e aplicação do TDD de forma mais acertiva.
Seria importante para um projeto possuir documentação e planejamento, acredito que seja bem importante.
Mesma resposta das anteriores.
Acredito que na fase de desenvolvimento é importante consultar os documentos, mas creio que eles são mais importantes na fase de análise e nessa etapas já devem estar bem assimilados.
Utilidade
Os desenvolvedores precisam saber o que precisa ser desenvolvido.
pelo grau de importância que avalio
Pois a documentação é gerada por outra área e precisamos provê-la de informações mínimas e suficientes para gerar a documentação necessária.
Familiaridade com o uso

Outra questão abordada foi se os documentos de desenvolvimento são suficientes para o ambiente de trabalho. A opção com mais votos foi concordo parcialmente com 56,5%, já a opção concordo totalmente e discordo totalmente obtiveram cada uma 17,4 %. A opção discordo totalmente obteve 8,7%. Apção indiferente não obteve votos.

Também foi questionado se os documentos são claros para o entendimento das necessidades do cliente. A opção concordo parcialmente obteve 52,2 % e foi a mais votada. A opção concordo totalmente e indiferente obtiveram 17,4 %. A terceira mais votada foi discordo totalmente, com 8,7% A opção discordo totalmente e indiferente obtiveram 4,3%.

A última questão abordada foi se o respondente teve problemas no desenvolvimento de software por falta de artefatos de software. A opção concordo parcialmente obteve 69,6 % e foi a mais votada. A opção concordo totalmente,

indiferente e discordo totalmente obtiveram cada uma 8,7%. A opção menos votada foi discordo parcialmente com 4,3%.

5.5 Artefatos na etapa de testes de software

Nessa etapa foram abordados se na empresa do respondente existe um setor de testes e quais artefatos gerados/usados no desenvolvimento de testes de software. Outro ponto abordado foi a importância dos mesmos e como eles podem afetar no processo.

O primeiro questionamento foi identificar se existe um setor de testes na empresa os respondentes trabalham. A maioria das respostas foi não, com 69,6% e sim foi 30,4%.

Outro questionamento foi quais documentos são usados no processo de testes de software. Os valores e percentuais são apresentados na figura 5. O valor com asterisco foi informado pelo respondente.

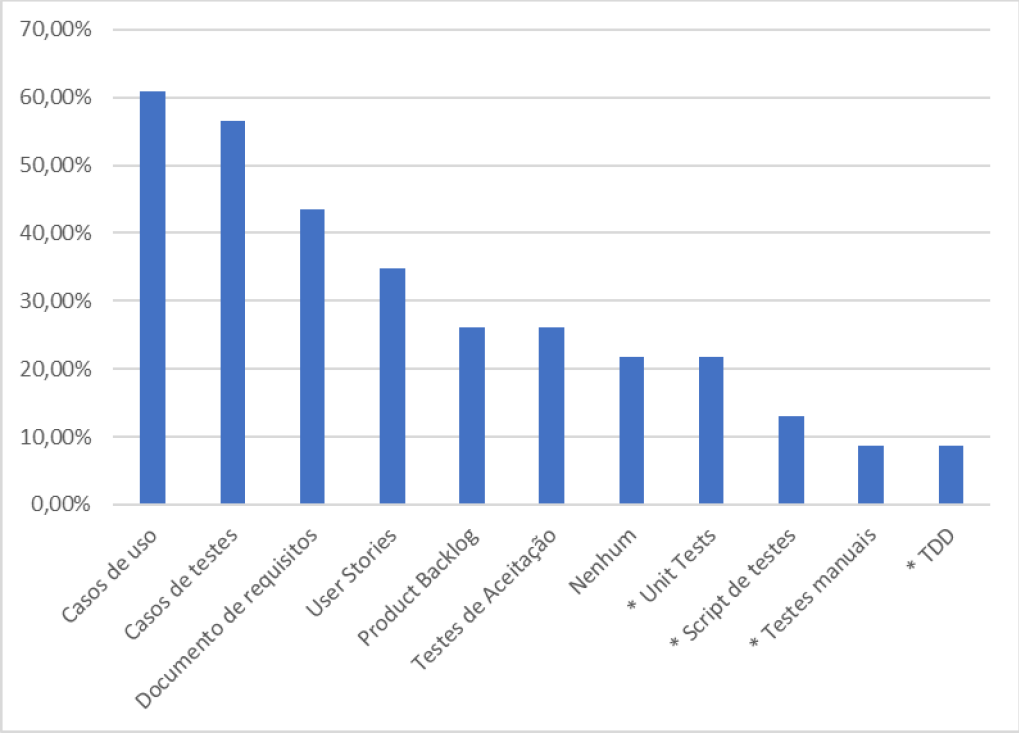

Figura 5 - - Documentos usados na etapa de teste de software

Outro importante questionamento foi medir a importância dos documentos de testes no processo de software os resultados foram os abaixo citados, na tabela 8.

Tabela 8 - Documentos usados na etapa de desenvolvimento de software e sua importância

	Muito importante	Importante	Neutro	Desnecessário	Muito desnecessário
Casos de uso	30,4%	30,4%	26,1%	0,0%	13,0%

Casos de testes	30,4%	39,1%	21,7%	0,0%	8,7%
Documento de requisitos	21,7%	34,8%	34,8%	0,0%	13,0%
User Stories	39,1%	21,7%	34,8%	0,0%	4,3%
Product Backlog	17,4%	30,4%	39,1%	0,0%	13,0%
Testes de Aceitação	30,4%	34,8%	30,4%	0,0%	4,3%

Outro ponto abordado foram os motivos das escolhas da pergunta anterior. Essa questão foi aberta foram listas as opções onde os respondentes escreveram um texto válido. As respostas obtidas foram as seguintes, conforme a tabela 9.

Tabela 9 – Respostas referente aos motivos das escolhas dos documentos na fase de design

Menos burocracia
Qualidade
A descrição das estórias para criar o fluxo de testes
Mesmo anterior
Ajuda a entregar algo com o máximo de qualidade.
Muita documentação
Possuir um melhor entendimento da regra de negócio
Possibilitam o teste
Detetar falhas
Apenas o básico, não somos focados em teste. Apesar de ser uma grande falha de nossa parte.
Auxiliam na validação dos testes.
Garantir que a entrega esteja de acordo com o requisitos levantados.
Este é um segmento de extrema importância, muitas vezes o QA deve conhecer o fluxo completo do sistema e impactos até mais que o desenvolvedor. Dessa forma os artefatos de teste devem ser claros.
Testes devem existir, somente assim se mantém a integridade do sistema.
Mesma resposta das anteriores.
Testar as regras negócio e o funcionamento de regras de domínio creio ser o mais importante nos testes.
Qualidade
Não sei dizer, pois nunca usei nada para testes.

Histórico
Considero os testes muito importante, preferencialmente automatizados, porém não dispensam os testes realizados por uma área específica, que analise o que foi implementado, faça os testes de unidade e de integração, garantindo que o sistema não tenha erros que comprometam suas funcionalidades críticas. Outro aspecto é que os testes também garantam que as novas funcionalidades ou correções de bugs estejam corretas e conforme o especificado.
Importância no processo de teste

Outra questão abordada foi se os documentos de testes são suficientes para o ambiente de trabalho. A opção com mais votos foi concordo parcialmente com 39,1%, já a opção concordo totalmente obteve 30,4 %. A opção indiferente obteve 17,4 % e discordo totalmente obteve 8,7%. A menos votada foi a opção discordo parcialmente com 4,3%.

A última questão abordada foi se os documentos são claros para testagem de software. A opção concordo parcialmente obteve 43,5 % e foi a mais votada. A opção indiferente foi a segunda mais votada e obteve 30,4 %. As opções concordo totalmente, discordo totalmente e discordo parcialmente obtiveram cada uma com 8,7%.

6. Conclusões

6.1 Principais Resultados

Com os resultados obtidos sobre o perfil do respondente foi possível embasar que a grande maioria são profissionais experientes e que estão fortemente ligados ao processo de desenvolvimento e análise de software. Também ficou evidente a experiencia com metodologias ágeis e a diversificação em vários tipos de projetos. Esse perfil corrobora a qualidade das informações obtidas.

Referente a etapa de análise de software foi possível constatar que não existe uma técnica majoritária na coleta de requisitos, mas sim diversas que são utilizadas com destaque para as previamente selecionadas, sendo esses as entrevistas presenciais, entrevistas não presenciais e código legado.

Sobre os documentos que são utilizados no processo de análise de software ficou destacado que regras de negócio, documento de requisitos e user Stories são os mais utilizados. Os documentos tradicionais da UML não se destacaram. Agora quando são relatados a importância dos mesmos, os documentos que obtiveram as melhores pontuações entre muito importante e importante foram as regras de negócio e documento de requisitos. Nesse aspecto alguns documentos que são pouco utilizados obtiveram pontuações altas, como diagramas de casos de uso e descrição de casos de uso.

Os motivos para essas respostas foram demonstrados na questão seguinte, que buscou entender o motivo das escolhas. Vários respondentes alegaram que utilizam metodologias ágeis e por isso buscam focar no desenvolvimento e não na documentação, alegando que é uma burocracia desnecessária. Mesmo sendo utilizados

poucos documentos, os respondentes alegaram que os mesmos são suficientes e claros para o processo de análise de software.

Sobre a etapa de design, segundo os respondentes os artefatos mais utilizados foram na grande maioria o protótipo de interface de usuário, user Stories, documentos de requisitos e regras de negócio, que são muito similares da etapa de análise. Já quando a questão sobre a importância dos artefatos, os resultados foram os mesmos com respostas para importante e muito importante. Novamente houve pouca participação dos documentos da UML, a exceção de uma pontuação média para a importância dos diagramas de casos de uso e descrição de casos de uso.

Sobre os motivos das escolhas, foi bastante destacado a importância do entendimento das telas e a busca de demonstrar uma previa ao usuário. Também foi relatado a busca por um processo com menos burocracia. Novamente nessa seção foi exposto que os documentos na fase de design são suficientes e claro no processo de software

Na etapa de desenvolvimento, apenas dois artefatos atingiram mais de 50% no uso de artefatos no processo de software, sendo esses documentos os requisitos e product backlog e com 43.5% dos casos de uso. As demais opções não atingiram 40%. Mas quando a questão foi a importância, houve uma maior valorização dos artefatos, com destaque para prototipagem, product backlog, documentos de requisitos e casos de uso. Nesse ponto, é possível observar que mesmo tendo um valor de importância, alguns documentos não são utilizados.

Sobre os motivos das respostas anteriores houve alguns respondentes que avaliaram que muitos documentos podem atrapalhar o desenvolvimento e que menos melhor. Porém também ficou evidenciados que os programadores precisam saber o que deve ser desenvolvido, bem como ter o mínimo suficiente para o desenvolvimento do processo de software.

Sobre a quantidade de artefatos serem suficientes no processo de software, a maioria respondeu positivamente, bem como são claros para o desenvolvimento de software. Porem a questão seguinte, que questionava se houve problemas no desenvolvimento por falta de artefatos de software, as respostas atingiram quase 80 %. O que reflete a importância dos artefatos.

A ultima etapa questionada, foi a de testes e nesse ponto chamou a atenção a quantidade de empresas que não tem um setor de testes, sendo esse praticamente 70%. Acredito que em função disso, nenhum artefato de teste atingiu 50% na questão que abordava quais artefatos são usados na testagem de software.

Quando foi questionado a importância desses documentos, quase todos atingiram mais de 50% para respostas importante e muito importante, a exceção do product backlog. Quando foi questionado os motivos, houve respostas que buscaram focar que devemos entender a regra de negócio para poder testar e também houve respostas para diminuir a burocracia.

Sobre se houve problemas na testagem pela falta de artefatos, foi possível observar que a grande maioria não encontrou problemas. Já quando questionado se os documentos são claro, a resposta atingiu um pouco mais de 50%.

6.2 Limitações

Em regras gerais, tomando com base todas as fases do processo de software e as respostas obtidas e a revisão da literatura, acreditasse que se pode chegar nas seguintes conclusões gerais.

Existe uma corrente muito forte que os processo de desenvolvimento deve ter o mínimo de documentação possível em todas as etapas. Até mesmo, não ter artefatos. As metodologias ágeis e sua grande disseminação são um dos grandes responsáveis disso. Uma vez que, ela vem em resposta aos métodos tradicionais, que buscavam focar no processo e não no produto.

Nesse ponto, é possível concluir que atualmente o processo de software vem utilizando cada vez menos artefatos e que os mais utilizados são: Documentos de requisitos, user Stories, regras de negócio e prototipagem. Esses documentos são mais voltados para identificar as necessidades do cliente. Os documentos mais tradicionais da UML estão sendo menos utilizados.

Outro ponto que ficou claro em todas as etapas são que os artefatos são suficientes para o entendimento da demanda do cliente e que sua quantidade não são afetando o processo de software. Mas não podemos ignorar que na etapa de desenvolvimento houve um numero significativo de problemas no desenvolvimento de software por falta de artefatos.

Apesar disso, muitos respondentes identificaram outros artefatos que seriam uteis no processo de desenvolvimento de software, porém não são utilizados. Acredito que um processo cada vez mais dinâmico de processo de software e mais passível de mudança pode ocasionar a diminuição de outros artefatos, uma vez que, com os sistemas sempre mudando as documentações ficam inúteis.

6.3 Trabalhos Futuros

Para trabalhos futuros, é possível avaliar alguns pontos para buscar um maior entendimento do tema.

- Identificar quais os motivos que os documentos da UML estão cada vez mais sendo menos utilizados;
- Um estudo mais aprofundado sobre o motivo da falta de uso de documentos e as mudanças constantes durante o projeto.
- O por que os envolvidos no processo de software entendem que a documentação é uma burocracia.

7. Referências

Ambler, S. W. (2018). An Introduction to Agile Modeling. From http://www.agilemodeling.com/essays/introductionToAM.htm

Barros, A. J. S, Lehfeld, N. A. S., (2007) Fundamentos da Metodologia Cientifica. Pearson

Crespo, A. N., Silva, O. J., Borges, C. A., Salvanio, C. F., Junior, M. T. A, Jino, M. (2000), Uma Metodologia para Teste de Software no Contexto da Melhoria de Processo.

Cunha, L. M. A. (2007) Modelos Rasch e Escalas de Likert e Thurstone na medição de atitudes.

Gerhardt, T. E., Silveira, D. T. (2009). Métodos de Pesquisa, 1ª ed.

Marconi, M. A. (2017) Fundamentos de metodologia científica. 8ª ed.

Pressman, R., Maxim, B. (2016). Engenharia de Software: uma Abordagem Profissional, 8ª edição

Sbrocco, J. H. T. C. e Macedo, P. C. (2012). Metodologias Ágeis Engenharia de Software Sob Medida. 1ª ed

Sommerville, I. (2011). Engenharia de Software, 9ª edição, Pearson Prentice Hall

Stettina, C. J., Heijstek, W. (2011), Necessary and Neglected? An Empirical Study of Internal Documentation in Agile Software Development Teams, Netherlands

Wazlawick, R. (2013). Engenharia de Software: Conceitos e Práticas. Edsevire, Brasil

Wagenaar, G., Helms, R., Damian, D., Brinkkemper, S. (2015) Artefacts in Agile Software Development

Wagenaar, G., Overbeek, S., Lucassen, Garm, Brinkkemper, Sjaak, Schneider, Kurt. (2018) Working software over comprehensive documentation – Rationales of agile teams for artefacts usage

APÊNDICE 1 - Questionário enviado

Seção 1

1 - Qual o seu nome?

2 - Qual é o seu cargo?

- Desenvolvedor
- Analista de Software
- Testador
- PO
- Gerente de projetos
- Outro

3 - Há quanto tempo você trabalha com desenvolvimento de software

- Até 2 anos
- 2 a 5 anos
- 5 a 10 anos
- 10 a 20 anos
- Mais de 20 anos

4 - Que tipo de projetos você já trabalhou?

- Web
- Desktop
- Web / Desktop
- Legado
- Produto novo
- Equipe distribuída

5 - Você trabalha com métodos ágeis?

- Sim
- Não
-

Seção 2 - Artefatos na análise de desenvolvimento de software

6 - Quais técnicas você utiliza para coletar requisitos?

- Entrevistas não presenciais (telefone, e-mail ou vídeo-conferencia)
- Entrevista presencial

- Código legado (Identifica uma solução já existente para levantar requisitos)
- Observação
- Outros

7 - Quais documentos você utiliza na análise do software?

- Classes de colaboração de responsabilidade
- Descrição de caso de uso
- Diagrama de atividade
- Diagrama de casos de uso
- Diagrama de classe
- Diagrama de robustez
- Documento de requisições técnicas
- Documento de requisitos
- Documento de restrições do sistema
- Glossário
- Mapa mental
- Mapeamento objeto-relacional (ORM)
- Modelo lógico de dados (LDM)
- Nenhum dos acima mencionados
- Regras de negócio
- User-Stories
- Outros

8 - Qual a importância que você vê dos seguintes documentos na análise de software?

	Muito importante	Importante	Neutro	Desnecessário	Muito desnecessário
Classes de colaboração de responsabilidade					
Descrição de caso de uso					
Diagrama de atividade					
Diagrama de casos de uso					
Diagrama de classe					
Diagrama de robustez					

Documento de requisições técnicas					
Documento de requisitos					
Documento de restrições do sistema					
Glossário					
Mapa mental					
Mapeamento objeto-relacional (ORM)					
Modelo lógico de dados (LDM)					
Regras de negócio					
User-Stories					

9 - Qual o principal motivo pela(s) escolha(s) dos documentos mais importantes da questão acima?

10 - Você acredita que os documentos de análise são suficientes no seu ambiente de trabalho?

- Concordo totalmente
- Concordo parcialmente
- Indiferente
- Discordo parcialmente
- Discordo totalmente

11 - Os documentos de análise são claros para entendimento das necessidades do cliente?

- Concordo totalmente
- Concordo parcialmente
- Indiferente
- Discordo parcialmente
- Discordo totalmente

3 - Artefatos no design de desenvolvimento de software

12 - Quais documentos você utiliza no design do software?

- Descrição de caso de uso

- Diagrama de atividades
- Diagrama de casos de uso
- Diagrama de classes
- Documento de requisitos
- Nenhum dos acima mencionados
- Protótipo de interface de usuário
- Regras de negócio
- UI Storyboards
- User Stories
- Outro

13 - Qual a importância que você vê nos seguintes documentos de design no software?

	Muito importante	Importante	Neutro	Desnecessário	Muito desnecessário
Diagrama de atividade					
Diagrama de classe					
Diagrama de casos de uso					
Descrição de caso de uso					
Documento de requisitos					
Protótipo de interface de usuário					
Regras de negócio					
User-Stories					
UI Storyboards					

14 - Qual o principal motivo pela(s) escolha(s) dos documentos mais importantes da questão acima?

15 - Você acredita que os documentos de design são suficientes no seu ambiente de trabalho?

- Concordo totalmente

- Concordo parcialmente
- Indiferente
- Discordo parcialmente
- Discordo totalmente

16 - Os documentos de design são claros para entendimento das necessidades do cliente?

- Concordo totalmente
- Concordo parcialmente
- Indiferente
- Discordo parcialmente
- Discordo totalmente

4 - Artefatos no desenvolvimento de software

17 - Quais artefatos você utiliza no desenvolvimento de software?

- Casos de uso
- Diagrama de atividades
- Diagrama de classes
- Diagrama de componentes
- Diagrama de estados
- Diagrama de objeto
- Diagrama de sequência
- Documento de requisitos
- Modelo de dados físico
- Nenhum dos acima mencionados
- Product Backlog
- Prototipagem
- Taskboard
- Testes de Aceitação
- Outro

18 - Qual a importância que você dos seguintes documentos no desenvolvimento de software

	Muito importante	Importante	Neutro	Desnecessário	Muito desnecessário
Casos de uso					
Diagrama de atividades					
Diagrama de classes					
Diagrama de componentes					
Diagrama de estados					
Diagrama de objeto					
Diagrama de sequência					
Documento de requisitos					
Modelo de dados físico					
Product Backlog					
Prototipagem					
Taskboard					
Testes de Aceitação					

19 - Qual o principal motivo pela(s) escolha(s) dos documentos mais importantes da questão acima?

- Concordo totalmente
- Concordo parcialmente
- Indiferente
- Discordo parcialmente
- Discordo totalmente

20 - Você acredita que tem a quantidade suficiente de artefatos (documentos) para o desenvolvimento de software?

- Concordo totalmente
- Concordo parcialmente
- Indiferente
- Discordo parcialmente
- Discordo totalmente

21 - Os documentos de requisitos geralmente são claros para o desenvolvimento de software?

- Concordo totalmente
- Concordo parcialmente
- Indiferente
- Discordo parcialmente
- Discordo totalmente

22- Você tem problemas no desenvolvimento de software por falta de artefatos de software?

- Concordo totalmente
- Concordo parcialmente
- Indiferente
- Discordo parcialmente
- Discordo totalmente

5 - Artefatos de testes no desenvolvimento de software

23 - Na sua empresa existe um setor de testes/QA?

- Sim
- Não

24 - Quais artefatos você utiliza na testagem de software?

- Casos de testes
- Casos de uso
- Documento de requisitos
- Product Backlog
- Testes de Aceitação
- User Stories
- Nenhum

25 - Qual a importância que você dos seguintes documentos de testagem de software

	Muito importante	Importante	Neutro	Desnecessário	Muito desnecessário

Casos de uso					
Casos de testes					
Documento de requisitos					
User Stories					
Product Backlog					
Testes de Aceitação					

26 - Qual o principal motivo pela(s) escolha(s) dos documentos mais importantes da questão acima?

27 - Você tem problemas na testagem de software por falta de artefatos de software?

- Concordo totalmente
- Concordo parcialmente
- Indiferente
- Discordo parcialmente
- Discordo totalmente

28 - Os documentos de testes são claros para a testagem de software?

- Concordo totalmente
- Concordo parcialmente
- Indiferente
- Discordo parcialmente
- Discordo totalmente