Estimativa de Software: Análise das Técnicas e Experiências de Profissionais *Freelancers*.

Gustavo Henrique Martins Vieira[1]

[1]Instituto de Ciências Exatas e Informática –
Pontifícia Universidade Católica de Minas Gerais (PUC MG)
Ed. Fernanda. Rua Cláudio Manoel, 1.162, Funcionários, Belo Horizonte – MG – Brasil

ghmvieira@sga.pucminas.br

Abstract. *One of the biggest challenges encountered in the software development process is projects without quality, within the planned projects. To deal with this, one can use software effort estimation techniques. This problem becomes even more critical when dealing with development by freelance professionals, as there is no technique specifically proposed for this type of professional to estimate their projects. This study aims to investigate how software estimation practices developed by professional freelancers. The work identifies which are the main estimation techniques to be used by developers with different levels of experience and the developer's experience regarding the impact on the accuracy of this estimation. The research methodology adopted mixed methods. The 1st. phase consisted of a software estimation survey applied to 12 former students of the Software Engineering course. The 2nd phase comprised an experiment to estimate the development time of a fictitious electronic retail system. The experiment involved 8 freelance developers. Their estimates were cross-checked with the expert opinion of an expert who employed the High-Level FPA technique. The majority of the developers that engaged in this study use only their own informal experience to estimate software. It was found that experience time does not influence the final accuracy of the estimate, and that it also leads developers to perform an estimate informally. It is also the use of estimation techniques, even simpler ones, helps in improving the estimation accuracy.*

Resumo. *Um dos maiores desafios encontrados no processo de desenvolvimento de software é concluir projetos com qualidade, dentro dos prazos e orçamentos previstos. Para lidar com isso, pode-se utilizar técnicas de estimativa de esforço de software. Esse problema se torna ainda mais crítico ao se lidar com o desenvolvimento por profissionais freelancer, pois não há uma técnica especificamente proposta para esse tipo de profissional estimar seus projetos. Este estudo tem por objetivo investigar as práticas de estimativa de software desenvolvido por profissionais freelancers. O trabalho busca identificar quais são as principais técnicas de estimativa a serem utilizadas por desenvolvedores com diferentes níveis de experiência e o quanto a experiência do desenvolvedor impacta na precisão dessa estimativa. A metodologia de pesquisa empregou métodos mistos. A 1a. etapa consistiu em um questionário sobre estimativa de software aplicado a 12 ex-alunos do curso de Engenharia de Software. A 2a. etapa compreendeu em um experimento de estimativa de tempo de desenvolvimento de um sistema fictício de varejo eletrônico. O experimento envolveu 8*

profissionais freelancers. Suas estimativas foram confrontadas com a especialista de uma especialista na área que empregou a técnica High-Level FPA A grande maioria dos desenvolvedores freelancer envolvidos no questionário e no experimento se baseia apenas na sua experiência informal para estimar seus projetos. Foi descoberto que o tempo de experiência não influencia na precisão final da estimativa, e que também, leva os desenvolvedores a realizarem a estimativa de maneira informal. Descobriu-se também, que o uso de técnicas de estimativa, mesmo as mais simples, ajuda na melhora da precisão da estimativa.

Bacharelado em Engenharia de *Software* - PUC Minas
Trabalho de Conclusão de Curso (TCC)

Orientador acadêmico (TCC I): Laerte Xavier - laertexavier@pucminas.br
Orientador do TCC II: Rodrigo Baroni - profrodrigobaroni@gmail.com

Belo Horizonte, 10 de junho de 2022.

1. Introdução

Um dos aspectos mais importantes de um projeto de software é estimar o quanto de esforço o software demanda. A estimativa de esforço é necessária para produzir uma estimativa de custos e um cronograma [Tsui et al. 2022]. Para isso, existem alguns modelos bastante conhecidos, tais como: Modelo de Custo Construtivo (COCOMO, do inglês *Constructive Cost Model*), a Análise por Pontos de Função (APF ou FPA, do inglês *Function Point Analysis*) e os Pontos de Casos de Uso (PCU ou UCP, do inglês *Use Case Points*). A métrica Pontos por Casos de Uso (PCU) foi proposta com o propósito de estimar recursos para projetos de software orientados a objeto, modelados por meio de especificação de Casos de Uso [Devmedia 2009]. A métrica Pontos de Função é uma medida de tamanho funcional de projetos de software, considerando as funcionalidades implementadas, sob o ponto de vista do usuário [Devmedia 2009]. O modelo básico COCOMO mede o esforço de desenvolvimento e custo em função do tamanho do software expresso em linhas estimadas de código (LOC, do inglês *Lines Of Code*). Neste trabalho, as técnicas de estimativa serão estudadas no contexto do desenvolvimento *freelancer* que são pessoas autônomas que têm uma associação de curto prazo baseada em tarefas com o empregador e, portanto, não fazem parte da força de trabalho da empresa [Gupta et al. 2020]

Um dos problemas mais enfrentados no desenvolvimento de software é conseguir estimar o projeto corretamente. A prática cotidiana mostra que muitas organizações de software ainda propõem custos de software irreais, trabalham dentro de cronogramas apertados e terminam seus projetos com atrasos e orçamentos superfaturados [Trendowicz and Jeffery 2014]. Diferente do desenvolvimento tradicional, não existe um modelo de estimativa de esforço de software especificamente proposto para estimar o esforço no contexto de desenvolvimento *freelancer* [Senevirathne and Wijayasiriwardhane 2020]. Diante disso, a estimativa se torna uma parte ainda mais crítica do desenvolvimento *freelancer*, pois não há uma forma do desenvolvedor prover uma estimativa precisa para o cliente. Por isso, o problema que este trabalho busca resolver **é a escassez de estudos sobre quais métodos de estimativa de esforço desenvolvedores *freelancers* de diferentes níveis de experiência devem usar**.

Para um projeto de software pequeno e bastante simples que envolve uma equipe de uma a três pessoas, estimar o esforço envolvido no desenvolvimento do projeto é relativamente fácil. Tanto os requisitos funcionais como os não funcionais do projeto são menores em número e complexidade, mesmo assim ainda não é uma tarefa trivial. Para sistemas grandes e complexos, capturar e entender os requisitos por si só pode ser desafiador. Estimar o esforço total e chegar a um cronograma de projeto confiável sob essas difíceis condições é uma das principais razões por trás de tantos fracassos em projetos de software. As estimativas e cronogramas de esforço imprecisos para sistemas grandes e complexos são muitas vezes extremamente otimistas, trazendo expectativas irreais em ambos, clientes e desenvolvedores desses sistemas [Tsui et al. 2022]. Portanto, é importante entender como desenvolvedores autônomos estimam seus projetos e investigar quais as melhores formas de se fazer essa estimativa sem uma ferramenta própria. Dessa forma, pode-se conseguir uma estimativa mais precisa e reduzir riscos dos projetos.

O objetivo deste trabalho é investigar as melhores práticas para se estimar o esforço/custo de um software *freelancer*. Para atingir esse objetivo são propostos os seguintes objetivos específicos: (i) descobrir como desenvolvedores autônomos de diferentes níveis de experiência estimam seus projetos; (ii) medir a precisão dessas estimativas; (iii) relacionar as estimativas com o tempo de experiência do desenvolvedor e ver se há alguma relação.

Espera-se descobrir quais são os modelos de estimativa mais utilizados entre os desenvolvedores autônomos e, a partir dessa descoberta, orientar os desenvolvedores quais são os modelos com maior taxa de sucesso a serem utilizados para cada nível de experiência. Com esses resultados, espera-se que este trabalho contribua com o processo de estimativa de custo/esforço de um software, tornando as estimativas mais precisas para desenvolvedores e clientes.

O restante deste trabalho está organizado em 4 seções. Na Seção 2, apresenta-se a fundamentação teórica. A Seção 3 contempla os trabalhos relacionados. Já a Seção 4 possui os materiais e métodos propostos.

2. Fundamentação Teórica

Nesta seção, são detalhados os principais conceitos e técnicas que estão relacionados à estimativa de esforço no desenvolvimento de software. São eles: o que é uma estimativa de software, o que são técnicas de estimativa e algumas dessas técnicas.

2.1. Estimativas de Software

Desde a década de 80, várias abordagens têm sido propostas e aperfeiçoadas com o objetivo de estimar o esforço empregado no desenvolvimento e o tamanho de um software. De acordo com Vazquez et al. (2003), o processo de estimativa de um projeto envolve, basicamente, três atividades:

1. Estimar o tamanho do produto a ser gerado;
2. Estimar o esforço empregado na execução do projeto;
3. Estimar a duração do projeto.

Para obter as respostas aos questionamentos iniciais de tempo e custo a partir da aplicação de um processo de estimativa, é necessário decidir a unidade para medir o tamanho do produto.

O processo de estimativa inicia-se a partir dos primeiros níveis de abstração dos requisitos do projeto. Para ser completo e eficiente, deve-se levar em consideração as experiências, os dados históricos de projetos passados e os recursos que cercam os projetos. A cada etapa realizada, as estimativas obtidas devem passar por um processo de aprovação antes de serem registradas na base histórica. Além de poderem ser utilizadas como fonte de informação para projetos futuros, podem ser utilizadas nas estimativas das etapas seguintes. Re-estimar deve ser uma atividade constante durante todo o ciclo de vida do desenvolvimento do software. Na etapa final do processo, com o produto concluído, as medidas reais de tamanho servem também para a validação e o aprimoramento de todo o processo de estimativa.

O termo estimativa é definido como uma medida aproximada da escala e carga do projeto necessária para o processo de estimativa de desenvolvimento. Após a verificação, recomendações técnicas e comerciais são feitas, planos detalhados e cronogramas são apresentados e o projeto é monitorado de forma eficaz.

2.2. Técnicas de Estimativas

Para se chegar à estimativa do esforço necessário para o desenvolvimento de software, existem muitas técnicas propostas na literatura. A seguir são apresentados algumas dessas técnicas, sendo elas: Pontos de Função, Pontos por Casos de Uso, *Planning Poker*, Comparação, Decomposição e Técnica de Avaliação e Revisão de Programas.

2.2.1. Pontos de Função

De acordo com o IFPUG (Grupo Internacional dos Usuários de Pontos de Função, do inglês *International Function Points Users Group*), Ponto de Função é uma unidade de medida padronizada usada para representar o tamanho de um software. O método de medição funciona quantificando os requisitos funcionais do software identificados, categorizando-os em cinco tipos: saídas, consultas, entradas, arquivos internos e interfaces externas. Uma vez que a função é identificada e categorizada em um tipo, ela é então avaliada para complexidade e atribuída uma série de pontos de função. Cada um dos requisitos funcionais do usuário mapeia para uma função de negócio do usuário final, como uma entrada de dados para uma entrada ou uma consulta de usuário para uma consulta. Essa distinção é importante porque tende a fazer com que as funções medidas em pontos de função mapeiem facilmente em requisitos orientados ao usuário, mas também tende a ocultar funções internas (por exemplo, algoritmos), que também exigem recursos para implementar. O número resultante é chamado uma contagem de Pontos de Função.

Devido ao propósito principal de um processo de estimativa ser o de fornecer informações que ajudem no planejamento e no controle dos projetos de software o mais cedo possível durante seu ciclo de vida, a utilização de pontos de função como unidade padrão de tamanho é a mais acertada. Isso se dá por uma série de motivos: softwares bem projetados ou com alto índice de componentização ou de reutilização de código têm seus indicadores baseados em LOC penalizados. Por exemplo, um sistema com 50.000 LOC, mas que exigiu um alto esforço para o desenvolvimento de uma arquitetura bem elaborada, irá apresentar uma estimativa final inferior à de outro projeto similar e de funcionalidade equivalente, porém com arquitetura menos trabalhada com 60.000 LOC.

Já com pontos de função, essa distorção não seria observada, uma vez que não existiria diferença relevante no tamanho estimado de ambos os projetos [Carvalho 2001].

A análise de pontos de função fornece aos usuários métricas objetivas e com visão de longo prazo, de forma que a relação entre usuários e desenvolvedores pode ser transformada por meio dos requisitos de métodos de medição de tamanho de função padronizados, e os dados podem ser interpretados para permitir para melhor entendimento. A influência é representada pelo ponto de função, e o usuário define o valor aproximado para o propósito de medição ou ponto de função a fim de entender a diferença de medição [Carvalho 2001].

2.2.2. Análise de Pontos de Função de Alto Nível

Medir pontos de função seguindo o processo padrão às vezes é longo e caro e só pode ser feito em fases avançadas do projeto. Para resolver este problema, vários métodos de estimativa inicial foram propostos. Dentre estes, o método "NESMA Estimated" é um dos mais utilizados e foi selecionado pelo IFPUG como o método oficial de análise de ponto de função inicial, sob o nome de APF de Alto Nível (do ingles *High-level FPA*) que também é chamada de APF Estimada. [Liu and Lavazza 2021]

De acordo com a Associação de Métricas de Software da Holanda (NESMA, do inglês *Netherlands Software Metrics Association*), o método de APF de Alto Nível é realizado da seguinte forma:

- Determine todas as funções de todos os tipos de função (ILF, EIF, EI, EO, EQ);
- Classifique a complexidade de cada função de dados (ILF, EIF) como Baixa e de cada função transacional (EI, EO, EQ) como Média
- Calcular a contagem total de pontos de função não ajustado

Assim, a única diferença dessa aproximação com o método detalhado de análise de ponto de função é que a complexidade não é determinada por função individual, mas por padrão.

2.2.3. Pontos por Casos de Uso

A técnica Pontos por Casos de Uso (PCU) foi proposta por Gustav Karner em 1993 com o propósito de estimar recursos para projetos de software orientados a objeto, modelados por meio de especificação de Casos de Uso. A métrica é de fácil aplicação e não requer muito tempo de treinamento ou experiência prática. No entanto, o PCU somente pode ser aplicado em projetos de software cuja especificação tenha sido expressa em casos de uso. Além disso, como não existe um padrão único para a escrita de uma especificação de caso de uso, diferentes estilos na escrita do caso de uso ou na sua granularidade podem levar a resultados diferentes na medição por PCU. [Devmedia 2009]

O método UCP fornece a capacidade de estimar as horas que um projeto de software requer a partir de seus casos de uso, o método UCP analisa os atores dos casos de uso, cenários, e vários fatores técnicos e ambientais e abstrai-os em uma equação. A equação UCP é composta de quatro variáveis:

- Pontos de caso de uso não ajustado (UUCP).

- O Fator de Complexidade Técnica (TCF). O Fator de Complexidade ambiental (ECF).
- O Fator de Produtividade (PF).

Assim sendo, a fórmula do UCP é UCP = UUCP * TCF * ECF * PF. [Clemmons 2006]

2.2.4. Outros Métodos de Estimativa

Nesta seção, são apresentadas outras técnicas de estimativa que podem ser usadas por desenvolvedores:

- ***Planning Poker***: O Planning Poker consiste-se em obter a estimativa por meio de um jogo de cartas, que, segundo Alves (2012), deve permitir que todos os membros da equipe de desenvolvimento (programadores, testadores, design e analistas) participem colocando a sua visão de complexidade, levando em consideração o fator tempo e esforço para pontuar um cartão e após juntos chegar a um denominador comum na equipe através de consenso.
  No Planning Poker, cada integrante tem a sua disposição um baralho de 13 cartas, numeradas em uma sequência similar a encontrada nos números de Fibonacci. As cartas contém os tamanhos de 0, ½, 1, 2, 3, 5, 8, 13, 20, 40 e 100 que serão atribuídos a um cartão, devem ser realizadas rodadas para obter a estimativa de um cartão que possui uma estória ou tarefa a desenvolver. As diferenças que surgirem durante as rodadas deverão ser mediadas por um coordenador. [Devmedia 2014]

- PERT (Técnica de Avaliação e Revisão de Programas, do inglês *Program Evaluation and Review Technique*) ou Técnica dos três pontos: O PERT é uma ferramenta utilizada no gerenciamento de projetos para calcular a duração de uma determinada atividade, um conjunto de atividades, ou mesmo de todo um projeto.
  Para calcular uma determinada atividade usando o PERT, você poderá utilizar-se de registros históricos de projetos semelhantes e opiniões dos especialistas no assunto. Sendo assim, considere:
  – Otimista (O) = É o cenário perfeito, onde tudo dá certo. – Pessimista (P) = É o pior cenário, onde tudo vai dar errado. – Mais Provável (M) = É um cenário razoável, onde a atividade acontecerá dentro da normalidade.
  A fórmula do cálculo PERT é: PERT = (Pessimista + 4 x Mais provável + Otimista) / 6. [Arruda 2016]

3. Trabalhos Relacionados

Nesta seção, são apresentados alguns trabalhos relacionados. Estes trabalhos envolvem a aplicação de técnicas de estimativa e avaliação em softwares desenvolvidos por *freelancers*, a análise de aplicações de técnicas de estimativa em manutenções de software em geral.

De acordo com Senevirathne and Wijayasiriwardhane (2020), a previsão de esforço leva em consideração vários fatores, que são diferentes no desenvolvimento *freelancer* e no desenvolvimento tradicional. Outra questão que atrapalha a previsão de esforço no desenvolvimento *freelancer* é que não há modelos de estimativa de

esforço especificamente propostos para estimar o esforço de desenvolvimento de software autônomo. Diante desses fatores, o trabalho destes autores busca estender o modelo de estimativa de esforço baseado em pontos de caso de uso para o desenvolvimento de software *freelancer*. Para isso, foram coletados detalhes de projetos de software de código aberto de vários desenvolvedores de software *freelancer* e foram calculados os pontos de caso de uso para cada um desses projetos com base nos diagramas de caso de uso. Foram testados quatro modelos de adaptação para os valores estimados: a mediana da analogia mais próxima, a média ponderada de classificação inversa da analogia mais próxima, a média da analogia mais próxima e a analogia mais próxima. Por fim, concluiu-se que a "Média ponderada de classificação inversa da analogia mais próxima"teve o melhor resultado. O artigo se relaciona a este trabalho ao analisar a estimativa de software no contexto do desenvolvimento *freelancer*. A diferença é que o objetivo do artigo citado é criar uma nova técnica de estimativa a ser usada pelos desenvolvedores *freelancers*, enquanto que o objetivo deste trabalho é entender como o tempo de experiência influencia na precisão da estimativa.

Para Iqbal et al. (2020), a terceirização de desenvolvimento de software está se tornando cada vez mais comum e alguns estudos confirmam que uma grande proporção dos projetos de terceirização de desenvolvimento de software não consegue obter sucesso. Os autores citam que de acordo com investigações sobre as falhas desses projetos, em muitos casos, projetos de terceirização de desenvolvimento de software falham devido a problemas associados ao processo de engenharia de requisitos. Devido a essas questões, o objetivo dos autores é identificar e hierarquizar os problemas comuns do processo de engenharia de requisitos no caso de terceirização de desenvolvimento de software. Para isso, a literatura contemporânea foi avaliada rigorosamente, os problemas enfrentados pelos profissionais foram identificados e três questionários foram organizados e aplicados para profissionais terceirizados experientes. Com os resultados obtidos, os autores chegaram à conclusão que a maioria dos problemas estão relacionados à comunicação e à gestão e coordenação. O artigo se relaciona a este ao analisar problemas em um contexto de terceirização do desenvolvimento de software. A diferença é que o artigo citado busca por problemas gerais ocorridos durante todo o ciclo de desenvolvimento de software, enquanto que este trabalho busca entender problemas relativos à estimativa de esforço no desenvolvimento de software.

De acordo com Jørgensen and Grov (2021), a qualificação do desenvolvedor e o tipo de contrato escolhido podem influenciar no resultado final do software. Com base nisso, o objetivo dos autores é testar duas hipóteses: a primeira é se o uso de *work-sample testing* (testes que verificam se o comportamento do colaborador e suas habilidades são consistentes com as características exigidas pela especificidade do trabalho) melhora a seleção de desenvolvedores de software qualificados; e a segunda é se o uso de contratos baseados em pagamento por hora leva a melhores resultados de projetos de software do que contratos de preço fixo. Para se chegar ao resultado, 57 desenvolvedores *freelancers* com experiência relevante e boas pontuações de avaliação de clientes anteriores foram convidados a concluir uma tarefa de avaliação de duas horas para se qualificar para um projeto de desenvolvimento de software. Assim sendo, 36 desenvolvedores concluíram a tarefa de *trialsourcing* com desempenho aceitável e, com base em um processo de alocação aleatória estratificada, foram solicitados a apresentar uma proposta baseada em pagamento por hora ou contrato de preço fixo. Os resultados encontrados mostram

que embora o uso de *trialsourcing* possa ter impedido a seleção de desenvolvedores com habilidades insuficientes, o desempenho na tarefa de *trialsourcing* dos desenvolvedores selecionados, em grande parte, não previu seu desempenho nos projetos. O uso de pagamentos por hora parece ter levado a custos mais baixos do que os contratos de preço fixo, mas não a uma melhoria no projeto. O artigo citado se relaciona a este trabalho ao avaliar o contexto de desenvolvedores *freelancers*. A diferença está no fato do artigo citado avaliar como o contrato impacta no preço final e na qualidade do software, enquanto este trabalho busca avaliar o quanto a experiência influencia na precisão da estimativa de esforço.

Para Shafiq et al. (2020), a terceirização de desenvolvimento de software é um dos paradigmas de desenvolvimento mais populares na indústria de software e há uma série de desafios associados a esse paradigma, incluindo os desafios relacionados ao processo de engenharia de requisitos. Com essas informações, os autores buscam identificar os fatores de sucesso associados ao processo de engenharia de requisitos no contexto do desenvolvimento terceirizado. Para se chegar ao resultado, 25 fatores de sucesso foram descobertos por meio de uma revisão sistemática de literatura e validados através de um questionário. Os autores também examinaram os desafios da engenharia de requisitos identificados em relação à sua relevância para diferentes tipos e tamanhos de organizações. O artigo citado se relaciona a este ao analisar o contexto do desenvolvimento terceirizado. Diferenciam-se no fato de que o artigo citado avalia aspectos relacionados à engenharia de requisitos enquanto este trabalho busca analisar aspectos relacionados à estimativa de esforço.

De acordo com Khan et al. (2021), os projetos de GSD (Desenvolvimento de Software Global, do inglês *Global Software Development*) compreendem vários direcionadores de custos críticos que afetam o custo geral do projeto e a sobrecarga orçamentária. Por isso, há a necessidade de ampliar o modelo de estimativa existente nesse contexto para reduzir os riscos associados à sobrecarga de custos. Motivado por isso, os autores visam ampliar o modelo de estimativa COCOMO com direcionadores de custo do contexto do desenvolvimento de software global, para obter estimativas mais eficientes. Para se chegar ao resultado, foram realizadas as seguintes atividades: identificação de direcionadores de custos; categorização de direcionadores de custos; formação de métricas; atribuição de valores; alteração da equação do modelo base. Além disso, a eficácia do modelo proposto foi validada por meio da avaliação de especialistas, estudos de caso e Magnitude das Estimativas Relativas. As estimativas obtidas são eficientes, quantificadas e abrangem aspectos do contexto em questão. Portanto, pode-se superar o risco geral do projeto GSD implementando o modelo. O artigo citado se relaciona a este ao estudar a estimativa de software no contexto do desenvolvimento global de software. Diferenciando-se no fato de que o artigo citado busca estender o modelo COCOMO para ser usado no contexto em questão, enquanto este trabalho busca entender como a experiência de um desenvolvedor *freelancer* afeta a precisão de sua estimativa.

4. Materiais e Métodos

O estudo proposto neste trabalho foi composto por três etapas: a primeira classificada como pesquisa qualitativa, a segunda como pesquisa quantitativa de natureza experimental e a terceira visou relacionar o resultado obtido nas duas primeiras etapas. O objetivo era, por meio de um questionário, descobrir como os desenvolvedores *freelancers* estimam seus projetos e, após isso, por meio de de métricas quantitativas, buscou-se identi-

ficar os impactos da experiência do desenvolvedor *freelancer* ao se estimar um projeto. Para tanto, foram avaliadas quais modelos são mais utilizados pelos desenvolvedores, seu tempo de experiência e o quão próximo a estimativa desse desenvolvedor foi da estimativa do especialista.

4.1. Procedimentos

Este trabalho foi composto por três etapas. Na primeira etapa, foi utilizado um questionário que teve como objetivo investigar quais são as técnicas de estimativa mais utilizadas pelos desenvolvedores *freelancers* e quais mais utilizadas por nível de experiência. Para a segunda, foi realizado um experimento composto pelas seguintes atividades: selecionar um pequeno software, fazer a estimativa dele utilizando a técnica de pontos de função de alto nível junto a um especialista e acompanhar os desenvolvedores *freelancers* enquanto eles estimam esse mesmo software utilizando uma técnica escolhida pelo autor. A etapa tem como objetivo coletar métricas relacionadas ao processo de estimativa de software, tais como: eficácia da estimativa (tempo estimado pelo desenvolvedor em relação ao tempo estimado pelo especialista) e facilidade de uso da técnica. Em seguida, os resultados do experimento são relacionados às respostas obtidas com o questionário, com o objetivo de identificar, para cada nível de experiência, qual a técnica mais utilizada e qual a que obteve melhores resultados.

As etapas foram realizadas com desenvolvedores *freelancers* e alunos do curso de Engenharia de Software da Pontifícia Universidade Católica de Minas Gerais (PUC Minas) que trabalham com desenvolvimento *freelancer*. O questionário foi livre para qualquer desenvolvedor *freelancer* que quiser responder, já o experimento foi executado com 8 desenvolvedores ao todo. Esses desenvolvedores foram separados em 3 grupos de acordo com seu tempo de experiência, sendo eles: 3 desenvolvedores com até 3 anos, 3 desenvolvedores com 3 a 6 anos e 2 desenvolvedores acima de 6 anos. O detalhamento do questionário encontra-se no Apêndice A.

Para a realização do experimento, inicialmente, os desenvolvedores informaram seu nível de experiência e quais técnicas costumam utilizar. Os desenvolvedores foram classificados de acordo com seu tempo de experiência. Então, o desenvolvedor fez a estimativa do software selecionado utilizando de uma técnica de sua escolha. Após a estimativa feita, ela foi comparada com a estimativa feita junto a um especialista para se avaliar o quão próxima/distante ela foi da sua estimativa.

4.2. Etapas da Pesquisa

A execução da pesquisa consistiu nas seguintes etapas:

1. Disponibilização e divulgação do questionário;
2. Seleção do software/requisito a ser estimado;
3. Estimativa do software/requisitos junto a um especialista;
4. Seleção dos desenvolvedores *freelancers* para participarem do experimento;
5. Realização do experimento;
6. Comparação dos resultados obtidos com o experimento com a estimativa realizada pelo especialista;
7. Comparação dos resultados do experimento com as respostas do questionário;
8. Escrita e revisão final do artigo.

4.3. Métricas e Avaliação

Conforme descrito na Seção 4.1, neste trabalho são realizadas três etapas. Para estas etapas, são utilizadas métricas relacionadas à experiência do desenvolvedor, à técnica que ele costuma utilizar e à eficácia dessas técnicas. Essas métricas são descritas da seguinte maneira:

- Experiência: É o tempo de experiência do profissional como desenvolvedor *freelancer*. Os desenvolvedores são divididos em grupos de acordo com o tempo de experência, os grupos são: desenvolvedores que tem até 3 anos de experiência no desenvolvimento *freelancer*, desenvolvedores com 3 a 6 anos de experiência e desenvolvedores que têm mais de 6 anos de experiência. Neste trabalho, utiliza-se essa métrica para classificar os desenvolvedores.
- Tempo estimado: É o tempo necessário para se desenvolver um software obtido com a estimativa. Neste estudo, essa métrica é utilizada com o objetivo de avaliar a eficácia da estimativa feita pelo desenvolvedor *freelancer*.

5. Análise dos Resultados

Nesta seção são analisados e explicados os resultados de cada uma das etapas da pesquisa.

5.1. Análise dos Questionários

A primeira etapa consistiu em aplicar o questionário, que foi enviado para redes sociais em grupos de desenvolvedores *freelancers* e para grupos de ex-alunos do curso de engenharia de software, que juntos tinham 25 alunos. Doze participantes o responderam durante todo o segundo semestre de 2022 e foram encontradas as seguintes respostas:

Em relação ao tempo de experiência dos desenvolvedores, 75% dos participantes possuem até 3 anos de experiência, 12,5% dos participantes possuem entre 4 e 6 anos de experiência, 12,5% dos participantes possuem mais de 6 anos de experiência.

Quanto a trabalhar sozinho ou com outros desenvolvedores, 75% dos participantes costumam trabalhar sozinhos e 25% dos participantes costumam trabalhar com outros desenvolvedores. Na amostra obtida, somente desenvolvedores iniciantes (até 3 anos) costumam trabalhar com outros desenvolvedores.

Os participantes deram as seguintes respostas sobre a quantidade de projetos desenvolvidos por ano: 50% dos participantes fazem até 3 projetos por ano, 37,5% dos participantes fazem entre 3 e 6 projetos por ano, dentre estes, 75% são iniciantes (até 3 anos de experiência), 12,5% dos participantes afirmaram não contar a quantidade de projetos.

Em relação a como definir as horas gastas ou técnicas usadas, 62,5% dos participantes afirmaram realizar a estimativa de maneira informal, tendo como base as experiências anteriores associadas a algum outro critério, como complexidade ou quantidade de telas. 12,5% dos participantes afirmaram utilizar pontos de função. 12,5% dos participantes afirmaram utilizar o *planning poker*. 12,5% dos participantes afirmaram utilizar pontos por caso de uso. Todos os participantes, que afirmaram utilizar alguma técnica são iniciantes (até 3 anos de experiência).

Nenhum dos participantes da amostra possui qualquer certificação em técnicas de estimativas, pois isto ainda é bastante raro no mercado brasileiro de *freelancers*.

Quanto à média de projetos superestimados ou subestimados, todos os participantes afirmaram que pelo menos 1 dos projetos desenvolvidos por ano foram gastas uma quantidade de horas diferente da estipulada inicialmente, 80% dos participantes afirmaram que mais de 50% dos projetos desenvolvidos por eles foram superestimados ou subestimados. Dentre estes, 83% afirmaram realizar a estimativa de maneira informal e 17% utilizam alguma técnica. 20% dos participantes afirmaram que menos de 50% dos projetos desenvolvidos por eles foram superestimados/subestimados. Dentre estes, todos afirmaram utilizar alguma técnica.

Os desenvolvedores deram as seguintes respostas sobre como registram as horas gastas em projetos anteriores: 62,5% dos participantes afirmaram não ter registro das horas gastas em projetos anteriores, 37,5% dos participantes afirmaram terem registro das horas gastas em projetos anteriores, 60% dos participantes que afirmaram realizar a estimativa de maneira informal com base em experiência não guardam registro das horas. 66% dos participantes que afirmaram usar alguma técnica para estimar os projetos não guardam registro das horas.

5.2. Análise dos Experimentos

A segunda etapa consistiu em estimar o desenvolvimento de um software cujos requisitos estão nos apêndices B e C. Além disso, nesta etapa também houve o acompanhamento dos desenvolvedores enquanto estimavam estes mesmos requisitos. Para a estimativa feita junto a um especialista, foi utilizada a técnica de High-Level FPA e foram encontrados os seguintes resultados:

- 20 requisitos ao todo;
- 4 Arquivos Lógicos Internos (ALI)
- 9 Entradas Externas (EE)
- 6 Consultas Externas (CE)
- 1 Saída Externa (SE)

Pela técnica do High-Level FPA, as funções de dados (ALI) recebem a complexidade baixa e as funções de transação (EE, CE e SE) recebem a complexidade média. Cada requisito recebe uma certa quantidade de pontos de acordo com a complexidade sendo:

- ALI: 7 Pontos
- EE: 4 Pontos
- CE: 4 Pontos
- SE: 5 Pontos

Com isso, chegou-se a quantidade de 97 pontos de função e foi aplicada uma quantidade de 6 horas por ponto, valor considerado como média pelo especialista. Assim, totalizaram-se 558 horas necessárias para se desenvolver estes requisitos. Os resultados detalhados podem ser encontrados na tabela 1.

Tabela 1. Contagem de Pontos de Função

Requisitos	Tipo Função	Complex.	Contagem PF	Estimativa	Esforço
Arquivo de Usuários	ALI	Baixa	7	Padrão	42
Incluir Usuário	EE	Média	4	Padrão	24

Alterar Usuário	EE	Média	4	Padrão	24
Consultar Dados Usuários	CE	Média	4	Padrão	24
Listar Usuário	CE	Média	4	Padrão	24
Excluir Usuário	EE	Média	4	Padrão	24
Login	SE	Média	5	Padrão	30
Arquivo de Produtos	ALI	Baixa	7	Padrão	42
Incluir Produto	EE	Média	4	Padrão	24
Alterar Produto	EE	Média	4	Padrão	24
Consultar Dados Produto	CE	Média	4	Padrão	24
Buscar Produtos	CE	Média	4	Padrão	24
Excluir Produtos	EE	Média	4	Padrão	24
Comunicação - Enviar Mensagem	EE	Média	4	Padrão	24
Comunicação - Listar Mensagens	CE	Média	4	Padrão	24
Arquivo de Mensagens	ALI	Baixa	7	Padrão	42
Adicionar Produto na Cesta de Compras	EE	Média	4	Padrão	24
Consultar Cesta de Compras	CE	Média	4	Padrão	24
Arquivo de Compras	ALI	Baixa	7	Padrão	42
Finalizar Compra	EE	Média	4	Padrão	24
Total Pontos			93	Total Horas	558

Já para as estimativas feitas com desenvolvedores, para o experimento que foi realizado na segunda quinzena de outubro e teve oito participantes, selecionados dentre desenvolvedores conhecidos que trabalham com desenvolvimento freelance, foram encontrados os seguintes resultados:

Em relação ao tempo de desenvolvimento e à técnica escolhida, todos os desenvolvedores com até 3 anos de experiência usaram alguma técnica. Todos os desenvolvedores com 3 a 6 anos de experiência usaram alguma técnica. Todos os desenvolvedores com mais de 6 anos de experiência realizaram a estimativa de maneira informal com base em experiências anteriores. 20% dos participantes usaram o *planning poker*. 20% dos participantes usaram o modelo de pontos do SCRUM (metodologia ágil de desenvolvimento). 20% dos participantes usaram a escala likert (escala de 5 pontos, geralmente usada em questionários) associada a uma quantidade de horas para cada item da escala. 20% dos participantes realizaram a estimativa de maneira informal com base em sua experiência;

Quanto à taxa de acerto ou de erro dos desenvolvedores e das técnicas, a técnica que mais se aproximou da estimativa do especialista (558) foi o *planning poker* com os resultados de 400 e 464. As estimativas mais próximas foram a de um desenvolvedor com 2 anos de experiência (mais próxima) e de um desenvolvedor com 4 anos de experiencia

(segunda mais próxima). As estimativas mais distantes foram a de um desenvolvedor com 5 anos de experiência (mais distante) e a de um desenvolvedor com 8 anos de experiência (segunda mais distante). A técnica com o pior desempenho foi a estimativa informal com base na experiência;

Os resultados detalhados podem ser encontrados na tabela 2.

Tabela 2. Resultados Experimento

Desenvolvedor	Anos como Desenvolvedor	Resultado Experimento	Técnica Usada
1	4	400	Planning Poker
2	8	155	Estimativa Informal Baseado em experiências anteriores
3	2	312	Modelo de pontos do SCRUM.
4	2	464	Planning Poker
5	7	760	Estimativa Informal Baseado em experiências anteriores
6	5	87	Modelo de pontos do SCRUM.
7	4	254	Escala Likert com uma quantidade determinada de horas para cada ponto da escala.
8	1	264	Escala Likert com uma quantidade determinada de horas para cada ponto da escala.

5.3. Relação dos Resultados

A terceira etapa da pesquisa, que consistiu em relacionar os resultados das duas primeiras etapas, foi executada na primeira quinzena de novembro e chegou-se aos seguintes resultados.

Todos os participantes do questionário que afirmaram usar alguma técnica de estimativa eram iniciantes. Já no experimento, todos os desenvolvedores iniciantes usaram alguma técnica. todos os participantes do questionário com mais de 6 anos de experiência afirmaram realizar a estimativa de maneira informal com base em experiências anteriores, já no experimento, todos os desenvolvedores com mais de 6 anos de experiência realizaram a estimativa informal. O mesmo não ocorreu em relação aos participantes do questionário e do experimento com tempo de experiência entre 3 e 6 anos.

Todos os desenvolvedores que responderam o questionário afirmaram não possuir nenhuma certificação na área de estimativa, o mesmo ocorreu entre os participantes do

experimento.

Nas respostas do questionário, 83% dos participantes que afirmaram terem mais de 50% dos projetos subestimados ou superestimados realizaram a estimativa informal, já no experimento, todos os desenvolvedores que realizaram a estimativa informal tiveram resultados bem distantes da estimativa de referência. O que pode indicar que realizar a estimativa de maneira informal, mesmo feito por desenvolvedores experientes e baseado em outros projetos parecidos, não é uma boa maneira de se estimar um software.

25% dos participantes do questionário afirmaram utilizar técnicas mais elaboradas como pontos de função ou pontos por casos de uso. Já no experimento, nenhum participante utilizou técnicas mais elaboradas.

Dentre os participantes que afirmaram ter menos de 50% dos projetos superestimados ou subestimados, todos eles afirmaram usar alguma técnica. Já no experimento, os melhores desempenhos foram obtidos pelas técnicas do *planning poker* e do modelo de pontos do SCRUM. O que pode indicar que o uso de técnicas de estimativa, mesmo as mais simples, pode ajudar a fazer uma estimativa mais precisa.

6. Discussão e Ameaças à Validade

Com base nos resultados encontrados, pode-se inferir que desenvolvedores *freelancers* não dão muita importância à etapa de se estimar o projeto, tendo em vista que nenhum dos participantes possuia certificação e cerca de 60% não guarda registro de horas de projetos anteriores. É possível afirmar também que mesmo para desenvolvedores mais experientes fazer a estimativa de maneira informal leva a projeções muito superestimadas ou subestimadas, logo, é melhor usar alguma técnica para se fazer as estimativas. Dentre as técnicas, o estudo da indícios de que dentre as técnicas mais simples, o *planning poker* tende a se sair melhor, logo, ele pode ser uma boa escolha para desenvolvedores que não têm certificação em técnicas mais completas estimarem seus projetos.

Tendo em vista que, a estimativa mais próxima foi de um desenvolvedor com 2 anos de experiência e que ela ficou 94 horas distante da estimativa do especialista, o que da cerca de 11 dias de trabalho, pode-se afirmar que o tempo de experiência não influencia na precisão final da estimativa e que desenvolvedores *freelancers* deveriam dar mais importância a esta etapa do desenvolvimento. Sendo assim, recomenda-se que os desenvolvedores, independente do tempo de experiência que possuem, guardem registros de horas de seus projetos desenvolvidos e que, mesmo desenvolvedores mais antigos, procurem não estimar seus projetos de maneira informal. Dessa forma, é possível garantir estimativas mais realistas para seus clientes.

Como ameaças à validade interna deste estudo, pode-se citar a pouca quantidade de participantes do questionário e do experimento. Como validade externa, o fato de que os desenvolvedores não estavam estimando um software que eles iriam desenvolver de verdade, logo, podem não ter levado tão à serio quanto em projeto real.

7. Conclusão e Trabalhos Futuros

Neste estudo, foi investigada a influência da experiência de um desenvolvedor *freelancer* na precisão de sua estimativa. Foi descoberto que estes desenvolvedores não dão muita importância a esta etapa. Além disso, pode-se afirmar que o tempo de experiência de um

desenvolvedor leva-o a ponderar as estimativas baseado em projetos anteriores, porém, essa estimativa informal tende a ser superestimada ou subestimada, e que, o tempo de experiência de um desenvolvedor não influencia na precisão final da estimativa. Este trabalho mostra também que o uso de técnicas de estimativa, mesmo as mais simples, faz com que mesmo desenvolvedores iniciantes possam fazer uma estimativa mais precisa.

Como trabalho futuro, sugere-se o treinamento de desenvolvedores para usar técnicas mais complexas como APF ou UCP para verificar se a precisão de suas estimativas seriam melhores. Dessa forma, poderia se chegar a conclusão se vale a pena ou não para um desenvolvedor buscar certificação em alguma dessas técnicas. Pode-se também utilizar a extensão da técnica UCP para o contexto de desenvolvimento *freelancer* [Senevirathne and Wijayasiriwardhane 2020], e ver se a adaptação desta técnica pode atender a estes desenvolvedores. Sendo assim, os desenvolvedores *freelancer* teriam uma técnica de referência para fazer suas estimativas.

8. Pacote de Replicação

O Pacote de Replicação deste trabalho encontra-se disponível em: `https://github.com/ICEI-PUC-Minas-PPLES-TI/plf-es-2022-1-tcci-5308100-pes-gustavo-henrique`

Referências

[Arruda 2016] Arruda, A. (2016). Calcular a duração de uma atividade — pert (estimativa de três pontos). `https://medium.com/@andreluis.arruda/calcular-a-dura%C3%A7%C3%A3o-de-uma-atividade-pert-estimativa-de-tr%C3%AAs-pontos-6cb35a09a1b3`.

[Carvalho 2001] Carvalho, A. M. (2001). *Introdução à engenharia de software*. Ed. da Unicamp.

[Clemmons 2006] Clemmons, R. K. (2006). Project estimation with use case points. *The Journal of Defense Software Engineering*, 19(2):18–22.

[Devmedia 2009] Devmedia (2009). Análise de ponto de função. `https://www.devmedia.com.br/analise-de-pontos-de-funcao/9146`.

[Devmedia 2014] Devmedia (2014). Scrum e planning poker: Análise de estimativa de software. `https://www.devmedia.com.br/scrum-e-planning-poker-analise-de-estimativa-de-software/31019`.

[Gupta et al. 2020] Gupta, V., Fernández-Crehuet, J. M., and Hanne, T. (2020). Freelancers in the software development process: A systematic mapping study. *Processes*, 8:1215.

[Iqbal et al. 2020] Iqbal, J., Ahmad, R. B., Khan, M., Alyahya, S., Nizam Nasir, M. H., Akhunzada, A., and Shoaib, M. (2020). Requirements engineering issues causing software development outsourcing failure. *PloS one*, 15(4):e0229785.

[Jørgensen and Grov 2021] Jørgensen, M. and Grov, J. (2021). A field experiment on trial-sourcing and the effect of contract types on outsourced software development. *Information and Software Technology*, 134:106559.

[Khan et al. 2021] Khan, J. A., Khan, S. U. R., Khan, T. A., and Khan, I. U. R. (2021). An amplified cocomo-ii based cost estimation model in global software development context. *IEEE Access*, 9:88602–88620.

[Liu and Lavazza 2021] Liu, G. and Lavazza, L. (2021). Early and quick function points analysis: Evaluations and proposals. *Journal of Systems and Software*.

[Senevirathne and Wijayasiriwardhane 2020] Senevirathne, D. S. and Wijayasiriwardhane, T. K. (2020). Extending use-case point-based software effort estimation for open source freelance software development. In *2020 International Research Conference on Smart Computing and Systems Engineering (SCSE)*, pages 188–194.

[Shafiq et al. 2020] Shafiq, M., Zhang, Q., Azeem Akbar, M., Alsanad, A., and Mahmood, S. (2020). Factors influencing the requirements engineering process in offshore software development outsourcing environments. *IET Software*, 14(6):623–637.

[Trendowicz and Jeffery 2014] Trendowicz, A. and Jeffery, R. (2014). *Software Project Effort Estimation*, pages 349–364.

[Tsui et al. 2022] Tsui, F., Karam, O., and Bernal, B. (2022). *Essentials of software engineering*. Jones & Bartlett Learning.

[Vazquez et al. 2003] Vazquez, C. E., SIMÕES, G. S., and Albert, R. M. (2003). Análise de pontos de função. *São Paulo: Érica*.

A. Questionário

Questionário usado para descobrir informações sobre os desenvolvedores freelancers. Informações sobre certificações em técnicas de estimativa, experiência com estimativa e outras relevantes à pesquisa. Ao final do projeto, o resultado do experimento será comparado com o resultado do questionário e ver se conferem.

- Há quanto tempo você trabalha com desenvolvimento freelance? você trabalha usualmente em projetos individuais ou em projetos com outros desenvolvedores *freelancers*?
- Qual a quantidade média de projetos você faz por ano?
- Como você define quantas horas serão gastas no projeto?
- Quais ferramentas/técnicas de estimativa de tamanho você utiliza?
- Você tem certificação em alguma técnica de estimativa de esforço de software?
- Qual a quantidade média dos projetos que você percebe que serão gastas uma quantidade de horas diferente da estipulada inicialmente?
- Você tem registros das horas que trabalhou em projetos anteriores?

B. Documento de Requisitos

Especificação dos requisitos a serem estimados durante o experimento. A partir deste documento de requisitos foi criado um diagrama de casos de uso para o especialista fazer a estimativa em pontos de função, para os demais desenvolvedores envolvidos no experimento, eles podem optar por fazer a estimativa usando este documento de requisitos ou o diagrama de casos de uso.

Projeto: Plataforma de Vendas

B.1. Descrição do Propósito do Sistema

Com o crescimento de vendas pela internet, podemos observar a necessidade de desenvolver um site mais interativo de vendas, no qual as pessoas que oferecem seus produtos possam descrevê-los detalhadamente. Logo, os interessados terã mais informações, podendo dialogar com o fornecedor e tirar suas dúvidas. Sendo assim, o usuário poderá expor seus produtos e comprar de maneira dinâmica.

B.2. Descrição do Sistema

O cliente deseja desenvolver um site de venda onde os usuários possam ter mais dinamicidade em vender e fazer compras, sem precisar estar em dois sites, logo nosso usuário poderá obter mais detalhes do produto que estará adquirindo conversando diretamente com seu fornecedor.

Assim que um usuário se cadastra no site, ele será automaticamente um comprador, tendo como opção ser ou não um vendedor. Caso queira se tornar um vendedor, basta cadastrar um primeiro produto. O usuário poderá conversar com outros clientes. Sobre o produto, ele terá uma categoria específica, descrição, valor, imagens. Cada produto será divulgado por meio de anúncios, sendo cobrado uma taxa mensal, que pode variar de acordo com a categoria.

O comprador escolherá os produtos que serão adicionados a uma lista(cesta) de pedido e escolherá a forma de pagamento (cartão ou boleto) e receberá o produto depois de confirmado o pagamento. Como o objetivo é ter mais praticidade, o cliente pode escolher entre receber em domicilio ou ele mesmo ir buscar.

O vendedor cadastrará seu produto escolhendo a categoria adequada e preenchendo todos os campos obrigatórios, que são descrição, valor, ano de compra do produto e imagens. Será oferecida uma lista de categorias padrão, o qual o vendedor escolherá de acordo com o produto. Caso não exista a mesma, o vendedor terá a opção de cadastrar outra.

Quando o comprador confirmar sua aquisição, o pedido é fechado e registrado, aguardando o pagamento para liberação dos itens.

B.3. Requisitos Funcionais

Requisitos funcionais do sistema podem ser encontrados na tabela 3.

Tabela 3. Requisitos Funcionais

Identificador	Descrição	Prioridade	Requisitos Relacionado
RF01	Os indivíduos que desejam divulgar seus produtos e comprar no site devem ter a possibilidade de descrevê-los de maneira detalhada afim de que todas as informações necessárias ao conhecimento do produto sejam difundidas, logo o comprador terá maiores informações do produto.	Alta	
RF02	O usuário deve ter a possibilidade de escolher entre os produtos existentes, permitindo a este visualizar todas as informações relevantes acerca desse, bem como armazená-lo na cesta de produtos do respectivo usuário.	Alta	RF01
RF03	Sempre que afixado pelos usuários os produtos que pretendem comercializar, as informações relativas a esses devem ser instantaneamente armazenadas pelo site, assim as pessoas que tenham pretensão de comprá-los obterão os dados atualizados do produto.	Média	RF01 RF02
RF04	O sistema deve possuir recursos que possibilitem a comunicação entre as partes envolvidas, de modo que se torne perceptível a alternativa de esclarecimento de questões que ficaram pendentes no que tange o produto.	Média	

RF05	Controlar Lista de Pedidos (Cesta de Produtos), o utilizador do site deve ter o poder de modificar as suas decisões de aquisição de produtos, podendo acrescentá-los ou retirá-los da sua cesta de produtos.	Alta	
RF06	Cadastrar no Site, aos utilizadores que não possuem cadastro no Site de Venda, deve ser permitido a esses efetuarem o cadastro afim de, terem acesso aos produtos lá disponibilizados, fornecendo a possibilidade a este de ser ou não um vendedor.	Alta	
RF07	Avaliar Preço da Compra com Todos os Encargos Adicionais. O utilizador deve poder visualizar o preço total da compra incluindo os encargos de transporte do item, podendo se for o caso de uma elevação no preço do produto, desistir da sua aquisição.	Média	RF05
RF08	Selecionar Modo de pagamento da compra. O usuário deve poder visualizar as alternativas existentes para que possa quitar o produto podendo ser por meio de: boleto ou cartão.	Alta	RF05 e RF08
RF09	Selecionar Opção de Entrega. O sistema deve apresentar os campos para escolha da opção de entrega podendo ser: entregue em domicílio ou mediante a própria busca do cliente no setor responsável por esta função.	Baixa	
RF10	Concluir Aquisição. Deve ser evidente ao usuário informações que demonstrem a integralização da aquisição do produto.	Alta	RF05, RF08 e RF09

RF11	Procurar Item. Deve estar contido nas páginas do site o recurso de pesquisa, que permita ao utilizador procurar pelo item específico de que necessita.	Baixa	RF12
RF12	Comunicar Aquisição do Item. Deve aparecer uma mensagem na tela do computador informando ao usuário a integralização do processo de aquisição do item.	Média	RF02, RF05, RF08
RF13	Manter Produto. O sistema deve manter as informações relativas ao produto tais como: tipo; descrição; valor e imagens do mesmo.	Alta	RF02 e RF03
RF14	Anunciar Produto. Uma vez que o produto foi cadastrado no Site de Venda, deve ser mantida a divulgação desse, pelo site, podendo também escolher por anúncios patrocinados mediante o pagamento de uma taxa pela prestação do serviço sujeito a alteração de valores conforme o tipo do produto.	Média	RF02, RF03 e RF04

B.4. Regras de Negócio

Regras de Negócio do sistema podem ser encontrados na tabela 4.

Tabela 4. Regras de Negócio

Identificador	Descrição	Prioridade	Requisitos Relacionados
RN01	O sistema deve permitir o acesso de qualquer pessoa ao site.	Baixa	
RN02	Somente pessoas cadastradas poderão fazer compras, vender, e ter e conversas com amigos.	Alta	RF01
RN03	O sistema deve permitir que o usuário adicione vários itens em sua cesta de compras, podendo cancelar a qualquer momento antes de efetuar o pagamento.	Média	RF01 3 RF02
RN04	Depois de cadastrado o usuário será automaticamente um comprador.	Média	

RN05	O usuário pode tornar-se um vendedor a partir do momento que cadastrar um primeiro produto	Alta	
RN06	O vendedor terá de cadastrar seu produto preenchendo todos os campos, deixando o produto mais detalhado possível, sendo as imagens importantes.	Alta	
RN07	Os produtos serão divulgados por meio de anúncio, e também pode utilizar anúncios patrocinados.	Baixa	RF05
RN08	Será cobrado do vendedor uma taxa por transação, e à medida que o mesmo ganhar reputação terá descontos sobre essa taxa.	Média	RF05 e RF08
RN09	Fazer Login, o utilizador deve necessariamente estar inserido no interior do site para que seja possível realizar a aquisição de produtos.	Alta	RF06
RN10	O usuário não pode postar itens com conteúdo vazio.	Média	RF01

B.5. Requisitos Não Funcionais

Requisitos Não Funcionais do sistema podem ser encontrados na tabela 5.

Tabela 5. Requisitos Não Funcionais

Identificador	Descrição	Categoria
RNF01	Caso o usuário opte pela forma de pagamento no cartão deve existir algum ou vários recursos no Site de Venda que transmita confiança a esse, que seus dados estão protegidos.	Segurança
RNF02	Para uma maior satisfação dos utilizadores ao procurar por produtos deve-se usar o recurso de pesquisa nas páginas que compõe o site.	Facilidade de Operação
RNF03	Deve-se ter uma estrutura de navegação fundamentada na solidez dos elementos que constituem o site de modo a manter uma coerência maior ao usuário durante a navegação por esse.	Consistência
RNF04	A edição das informações relativas aos produtos a serem vendidos estão restritas ao usuário identificado no cadastro como vendedor que realizou o registro dos itens.	Segurança

RNF05	A projeção da interface de navegação deve ser planejada de modo que o utilizador do site consiga percorrer as páginas que o compõem sem grandes transtornos.	Facilidade de Operação
RNF06	Deve existir um espaço no site preferivelmente na página central destinado à exposição das informações mais importantes as quais ele agrega.	Direcionamento do Usuário
RNF07	A interface de navegação do site deve ser estável mesmo em tamanhos de tela diferentes.	Portabilidade

C. Diagrama de Casos de Uso

Especificação dos casos de uso a serem estimados pelos desenvolvedores e pelo especialista. Casos de usos criados a partir do documento de Requisitos.

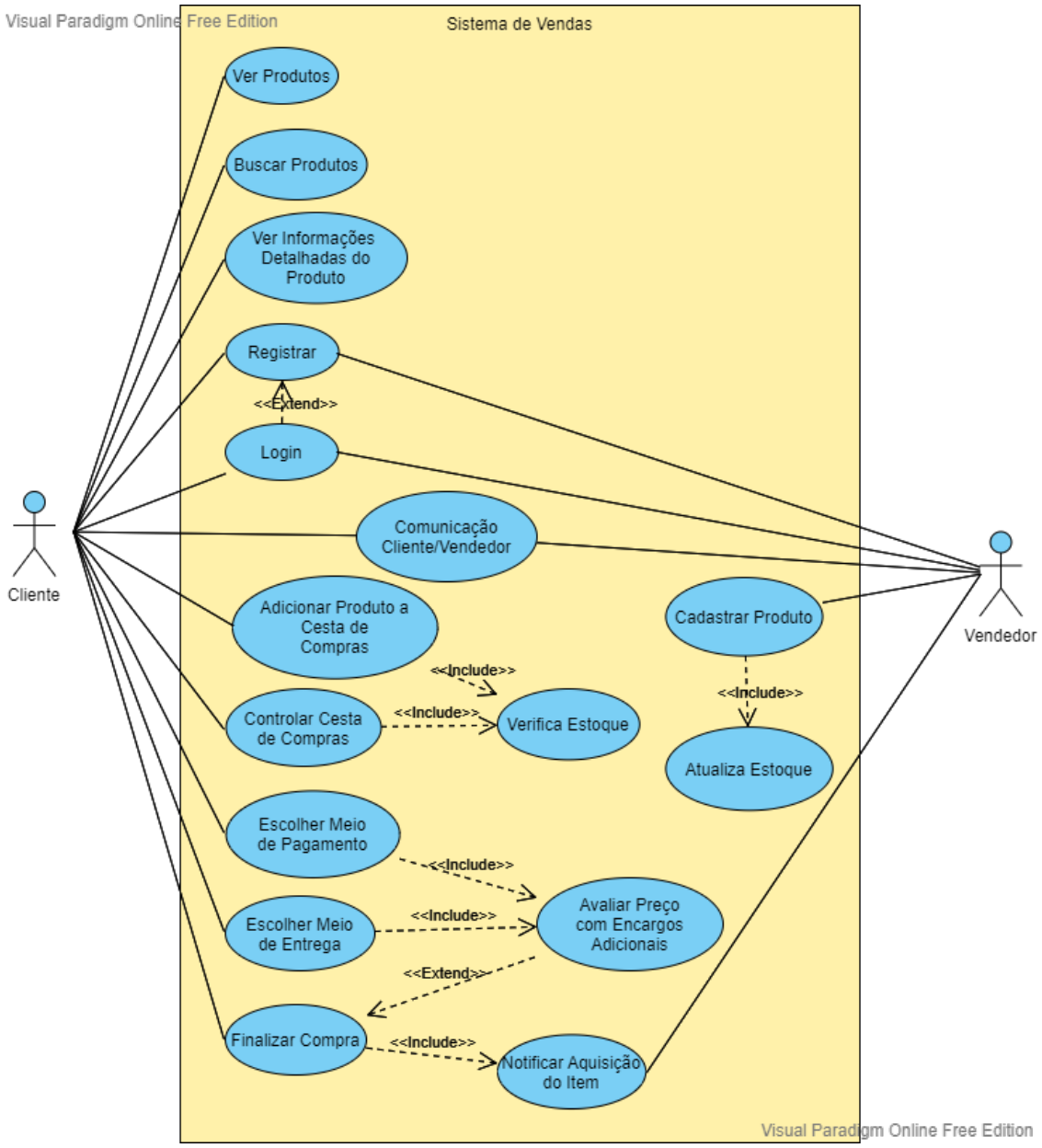

Figura 1. Diagrama de Casos de uso