UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
UNIDADE ACADÊMICA DE SISTEMAS E COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

DALES EWERTON LOPES FRAGOSO

CÁLCULO HIPOTÉTICO UNIVERSAL TÉCNICO ESTIMATIVO – C.H.U.T.E

CAMPINA GRANDE - PB

2019

DALES EWERTON LOPES FRAGOSO

CÁLCULO HIPOTÉTICO UNIVERSAL TÉCNICO ESTIMATIVO – C.H.U.T.E

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Professor Dr. Kyller Costa Gorgônio.

CAMPINA GRANDE - PB

2019

F811c Fragoso, Dales Ewerton Lopes.
Cálculo hipotético universal técnico estimativo – C.H.U.T.E. / Dales Ewerton Lopes Fragoso. – 2019.

10 f.

Orientador: Prof. Dr. Kyller Costa Gorgônio.
Trabalho de Conclusão de Curso - Artigo (Curso de Bacharelado em Ciência da Computação) - Universidade Federal de Campina Grande; Centro de Engenharia Elétrica e Informática.

1. Desenvolvimento de software. 2. Desenvolvimento ágil de software. 3. Laboratório de Sistemas Embarcados e Computação Pervasiva - UFCG. 4. Estimativas em desenvolvimento de software. 5. Root Mean Squared Error -RMSE. I. Gorgônio, Kyller Costa. II. Título.

CDU:004(045)

Elaboração da Ficha Catalográfica:

Johnny Rodrigues Barbosa
Bibliotecário-Documentalista
CRB-15/626

DALES EWERTON LOPES FRAGOSO

CÁLCULO HIPOTÉTICO UNIVERSAL TÉCNICO ESTIMATIVO – C.H.U.T.E

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

Professor Dr. Kyller Costa Gorgônio
Orientador – UASC/CEEI/UFCG

Professora Dra. Francilene Procópio Garcia
Examinadora – UASC/CEEI/UFCG

Professor Dr. Tiago Lima Massoni
Examinador – UASC/CEEI/UFCG

Trabalho aprovado em: 02 de julho de 2019.

CAMPINA GRANDE - PB

Cálculo Hipotético Universal Técnico Estimativo - C.H.U.T.E

Um estudo sobre estimativas de tarefas feitas por programadores iniciantes

Dales Fragoso
Departamento de Sistemas e Computação,
Universidade Federal de Campina Grande,
Campina Grande, Paraíba, Brasil,
dales.fragoso@ccc.ufcg.edu.br

Kyller Costa Gorgônio
Departamento de Sistemas e Computação,
Universidade Federal de Campina Grande,
Campina Grande, Paraíba, Brasil,
kyler@dsc.ufcg.edu.br

RESUMO

Um grande problema no início de carreira de todo desenvolvedor é saber quanto tempo ele levará para concluir uma tarefa. Na falta de experiência, não sabemos quais critérios um programador iniciante usa na hora de estimar a duração do seu trabalho. Contudo, analisando mais de perto como esse processo é feito podemos separar as boas das más práticas e auxiliar, tanto os desenvolvedores iniciantes quanto seus gerentes, a realizar estimativas mais efetivas.

Palavras-chave

Estimativas, Desenvolvimento Ágil, Análise Exploratória Descritiva.

1. Introdução

O interesse em estudar melhores maneiras de se estimar o esforço necessário para a realização de tarefas em projetos tem sido uma constante na engenharia de software [4]. Grandes teóricos da Ciência da Computação, incluindo Barry Boehm [5], se debruçaram sobre o tema ao propor as mais variadas técnicas e métodos com o objetivo de quantificar o esforço em tarefas e projetos computacionais.

Um desses métodos, o Scrum, é um processo de desenvolvimento de software que tem atraído uma boa parcela dos estudos da área nos últimos anos [1]. Tal modelo requer, em vários estágios do ciclo de desenvolvimento, que os membros envolvidos se reúnam para realizar as estimativas necessárias com o objetivo de estabelecer, entre outras coisas, a quantidade de tarefas a serem desenvolvidas em uma iteração (Sprint) e qual o esforço necessário para a realização de cada uma das tarefas planejadas.

Contudo, as técnicas, modelos, e principalmente a experiência necessária para realizar tais estimativas raramente fazem parte do repertório de conhecimentos dos programadores em seus primeiros projetos. Propomos então, uma análise e uma breve discussão sobre as tentativas de realização de estimativas por programadores ainda em período de graduação.

2. Fundamentação teórica

Processo de desenvolvimento de software é um termo usado para representar um conjunto de atividades parcialmente ordenadas que deve ser seguido com o objetivo de produzir um determinado software. O Scrum é um dentre os vários processos disponíveis para o desenvolvimento de software que tem despertado grande interesse por parte da academia e da indústria nos últimos anos. Nele, os projetos são divididos em ciclos, chamados Sprint, cuja duração pode variar tipicamente entre uma semana e um mês. Durante uma Sprint um conjunto de atividades devem ser executadas com o objetivo de produzir um incremento funcional do produto em desenvolvimento.

Dentre as diversas atividades executadas durante uma Sprint, estamos interessados em uma conhecida como Sprint Planning Meeting. Nessa reunião é discutido o incremento do produto que deve ser concluído até o fim da Sprint. De posse dessa informação, a equipe de desenvolvimento deve, durante essa reunião, dividir o incremento em

pequenas tarefas e realizar uma estimativa do tempo necessário para a conclusão de cada tarefa.

Uma das diversas técnicas utilizadas para realizar as estimativas necessárias dentro do Scrum é o Planning Poker. Tal técnica consiste inicialmente na realização de uma estimativa individual da quantidade de esforço necessário para a conclusão de cada uma das tarefas. A partir do resultado das estimativas individuais para uma tarefa, a equipe debate sobre os motivos pessoais da escolha do valor estimado por cada membro e entra em acordo sobre qual valor deve ser mantido como estimativa final para a tarefa.

Em 2011, George Stark [2] aplicou modelos clássicos de estimativa de esforço em projetos da IBM para estudar como sua acurácia varia de acordo com os parâmetros de cada projeto. Para lidar de forma rápida e eficaz com os dados disponíveis, ele utilizou algoritmos de aprendizado de máquina, técnica que viria a ser amplamente utilizada nos estudos da área a partir desta data.

Dantas et al. [1], em 2018, nos forneceu uma boa base para entender o estado da arte de estimar esforço no Desenvolvimento Ágil. Ao filtrar e analisar publicações sobre estimativa de esforço no desenvolvimento de software publicadas entre 2014 e dezembro de 2017, ele identificou o crescimento da utilização de técnicas inteligentes baseadas em dados históricos como suporte aos métodos de estimativas de esforço. Seu estudo também conclui que a métrica Story Points, aliada a técnica de Planning Poker, tem sido usada como base para as estimativas segundo a grande maioria da literatura analisada.

3. Metodologia

3.1 **Objetivos e perguntas**

Nesse estudo, buscamos compreender as dificuldades que desenvolvedores de software enfrentam ao lidar com estimativas de tarefas no início de suas carreiras.

Para isso buscamos responder como o tipo de tarefa ou seu contexto impactaram as estimativas avaliadas.

Outra questão de interesse é entender como diferentes projetos, que possuem diferentes parâmetros, se relacionam com a quantidade de erros cometidos durante a realização de estimativas.

3.2 **Contexto do estudo**

Foram analisados dados de 12 projetos desenvolvidos no Laboratório de Sistemas Embarcados e Computação Pervasiva (Embedded), que faz parte do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande (UFCG). Todos os projetos adotaram Scrum como processo de desenvolvimento e a técnica de Story Points aliada ao Planning Poker como principal recurso para a realização das estimativas. Os desenvolvedores envolvidos nos projetos eram alunos regularmente matriculados no curso de Ciência da Computação da UFCG localizados entre o segundo e oitavo período.

Todos os dados analisados foram extraído de forma anônima a partir de uma base de dados coletados ao longo de toda a execução dos projetos por alunos de doutorado do Intelligent Software Engineering (ISE). Nenhum tipo de questionário ou formulário foi aplicado durante a realização do estudo.

3.3 **Procedimento executado**

Inicialmente foram agrupados dados sobre 1267 tarefas de desenvolvimento de software realizadas por alunos nos projetos selecionados. Em seguida, foi realizada uma filtragem para remover tarefas com dados inconsistentes e alguns atributos irrelevantes. Após a filtragem, restaram um total de 721 tarefas. A tabela 3.1 mostra a distribuição de dados coletados por projeto.

Projeto	**N ° de tarefas**
Sistema de Gestão	206
Turmalina NFRec	117
Sistema de Seleção	86
Biblioteca UFCG	78

Kaizen Light	52
RU UFCG	45
Team Formation	35
WebBN	33
Kaizen	31
Dona Plantinha	18
SAM	16
Waterbending	4

Tabela 3.1 - Distribuição de dados por Projeto

Cada tarefa, após o processamento, contava com os seguintes dados:

- Esforço estimado em horas
- Esforço real em horas
- Categoria
- Operação
- Tarefa

Com o objetivo de normalizar os dados, as tarefas foram divididas em três categorias: cadastro; autenticação; e gerenciamento.

A categoria cadastro possui tarefas cujo objetivo principal é o de introduzir novas funcionalidades no sistema em desenvolvimento. Já a categoria de autenticação contém tarefas relacionadas a mecanismos de permissão e controle de acesso. Por fim, a categoria gerencial contém tarefas que de alguma forma facilitam a administração do sistema.

As tabelas 3.2, 3.3 e 3.4 exibem as tarefas contidas em cada uma das categorias. É importante observar que as tarefas filtradas são comumente atribuídas à desenvolvedores iniciantes.

Tipo da tarefa	**N° de ocorrências**
Modificar inserção de dados	197
Recuperar dados	180
Inserir dados	148
Atualizar dados	49
Remover dados	30

Tabela 3.2 - Tarefas da categoria Cadastro

Tipo da tarefa	**N° de ocorrências**
Criar conta	25
Fazer login	23
Recuperar senha	19
Acessar pela primeira vez o sistema	14
Validar permissão de usuário	7

Tabela 3.3 - Tarefas da categoria Autenticação

Tipo da tarefa	**N° de ocorrências**
Notificar via aplicação	16
Notificar via email	10
Visualizar dashboard	3

Tabela 3.4 - Tarefas da categoria Gerencial

3.4 **Método de pesquisa**

Para verificar se realmente os parâmetros estudados por hipótese podem ter relação com os erros cometidos nas estimativas, é necessário, inicialmente, definir o erro.

3.4.1 Definição de erro

Para fins de análise, definimos o erro de uma estimativa como sendo $E = Tr - Te$, onde E é o erro de estimativa de uma tarefa, Tr é o tempo real do desenvolvimento da tarefa e Te é o tempo estimado pelo desenvolvedor para a realização da tarefa.

A partir da definição, temos que os erros podem ser:

- iguais a zero; para estimativas totalmente corretas.
- maiores que zero; para tarefas que levaram mais tempo que o estimado.
- menores que zero; para tarefas que levaram menos tempo que o estimado.

De posse desses valores, agrupamos os erros nos seguintes grupos a fim de distingui-los com maior facilidade:

- Estimativas corretas;
- Estimativas com erros de até duas horas
- Estimativas com erros superiores à duas horas
- Estimativas com erros negativos

A figura 3.1 mostra os grupos criados e suas proporções.

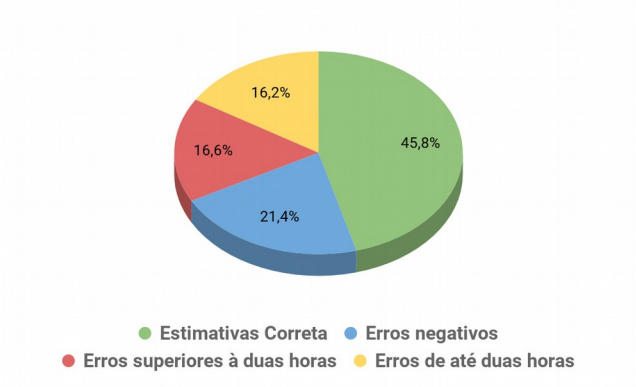

Figura 3.1 - Agrupamento dos Erros

Foram consideradas como estimativas corretas aquelas que apresentaram um erro, negativo ou positivo, inferior a 30 minutos.

É importante ressaltar que as tarefas executadas nos projetos avaliados foram planejadas para serem produzidas em no máximo 4 horas. O motivo disso está no fato de que os desenvolvedores que participaram do estudo dedicavam 20 horas semanais ao projeto, sendo 4 horas o valor médio de trabalho diário para eles.

Portanto, a classe de erros superiores à duas horas foi considerada durante a análise dos resultados como uma classe com altos valores de erro, visto que tal classe representa erros superiores a 50% do tempo máximo de realização da tarefa.

3.4.2. Classificação dos Resultados

Ao comparar os resultados e definir quais parâmetros são de fato impactantes diversas vezes fizemos uso de uma métrica de avaliação chamada de *RMSE.* O *RMSE (Root Mean Squared Error)* é uma medida estatística que calcula a raiz quadrada da média dos valores do erro elevados ao quadrado.

Com o objetivo de classificar quão forte é a relação entre o projeto e o erro associado às estimativas, na sessão 5, construímos uma regressão linear. A regressão linear é uma equação utilizada para se estimar o valor esperado de uma variável *y*, conhecida como variável dependente, dado valores definidos para um conjunto de outras variáveis, chamadas de variáveis independentes.

É importante observar que retiramos do conjunto de dados utilizados na regressão linear os projetos que possuíam menos de 30 tarefas nos dados filtrados, pois tais conjuntos não seriam representativos e causariam um impacto negativo aos resultados.

4. Relação entre erro e tarefa

A partir dos valores definidos para o erro tentamos encontrar relações entre ele e alguns parâmetros do desenvolvimento da tarefa.

4.1 Relação entre erro e a definição da tarefa

O primeiro parâmetro a ser analisado é a definição da tarefa. Procuramos por alguma relação entre tarefas e o erro relacionado a mesma. Para tanto, utilizamos o *RMSE* como métrica de avaliação com o objetivo de definir as tarefas com um maior erro de estimativas.

A tabela 4.1 exibe as 10 tarefas com os maiores erros seguindo essa definição.

Tarefa	**RMSE**
Expandir informações de item da listagem	4.462
Fazer validação de dados no cliente	3.605
Criar middlewares para validação do token	3.591
Alterar menu do sistema	3.553
Modificar tela inserção	2.812
Validar e-mail e senha informados	2.397

Recuperar dados do banco de dados	2.343
Modificar validação de dados no cliente	2.236
Criar ordenação para listagem	2.160
Criar tela para cadastro de nova senha	2.121

Tabela 4.1 - 10 tarefas com maior RMSE

Note que a tarefa com maior RMSE, 'Expandir informações de item da listagem', bem como outras tarefas localizadas entre as 10 com maior RMSE, como 'Alterar menu do sistema' e 'Modificar tela inserção' são tarefas que requerem algum tipo de refatoramento de código, seja para inserir funcionalidades ou modificar itens em módulos já existentes.

Segundo Arcelli Fontana [6], quanto mais o código fonte do projeto se torna complexo, mais difícil se torna a sua manutenção e modificação. Portanto, é de se supor que programadores iniciantes tenham dificuldades para estimar esse tipo de tarefas e podemos perceber que de fato isso acontece na prática quando olhamos para a *tabela 4,1*.

Além das tarefas que envolvem refatoramento, as que necessitam de algum tipo de validação de dados, como 'Criar middlewares para validação do token' e 'Validar e-mail e senha informados', também parecem ter sido frequentemente subestimadas nos projetos estudados.

4.2 **Palavras-chave relacionadas aos erros**

Dado as descrições das tarefas, procuramos encontrar as palavras-chave que mais ocorrem em tarefas que continham altos valores de erros nas estimativas, ou seja, aquelas que pertenciam a classe com valores superiores a 2 horas.

A visualização presente na figura 4.1 nos mostra as palavras que mais ocorreram em tarefas cujos erros de estimativas foram superiores a duas horas. O tamanho das palavras na imagem é proporcional a quantidade de vezes que ela ocorreu.

A partir dela podemos identificar pontos presentes em tarefas que podem requerer uma atenção especial por parte dos programadores iniciantes no momento de estimar o tempo necessário para concluir a tarefa.

Figura 4.1 - Palavras que mais ocorrem em tarefas com erros superiores à duas horas

Podemos ver que as duas palavras com maior frequência no grupo de tarefas com erros de estimativas maiores que duas horas são 'tela' e 'dados'. Esse problema pode estar relacionado ao fato de que programadores, no início de seus estudos, tendem a ter um maior contato com lógica de programação e algoritmos. Justamente por isso é comum que tais programadores careçam de conhecimentos na àrea de interface gráfica ou mesmo bancos de dados.

4.3 **Relação entre erro e a categoria**

O último parâmetro que queremos analisar dentro da relação de erro e tarefa são as categorias. Existe relação entre a categoria em que a tarefa se enquadra e a classe do seu erro?

A figura 4.2 mostra um diagrama de caixa da relação entre as categorias, no eixo y, e o valor de erro da estimativa, no eixo x.

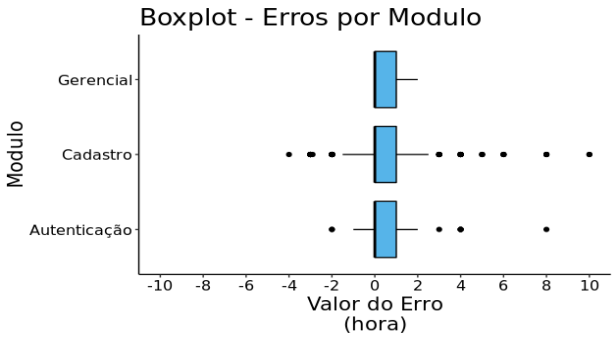

Figura 4.2 - Diagrama de Caixa do Erro x Categoria

Podemos perceber que, contrariando a hipótese inicial, as tarefas das categorias apresentam distribuições semelhantes, o que nos leva a crer que não existe nenhuma forte relação entre o erro e a categoria.

Contudo, algumas tarefas, principalmente na categoria Cadastro, se mostram bem distantes do erro médio das estimativas e merecem um olhar mais cuidadoso, que será dado na seção 6.

5. Relação entre erro e projeto

Diferentes projetos possuem diferentes domínios e diferentes graus de dificuldade associados. Estamos interessados em saber se tais divergências têm impacto sobre as estimativas de tarefas realizadas pelos programadores analisados.

Para responder essa questão, começamos selecionando seis projetos para comparar seus erros de estimativa de maneira mais precisa através de um diagrama de caixa exibido na Figura 5.1.

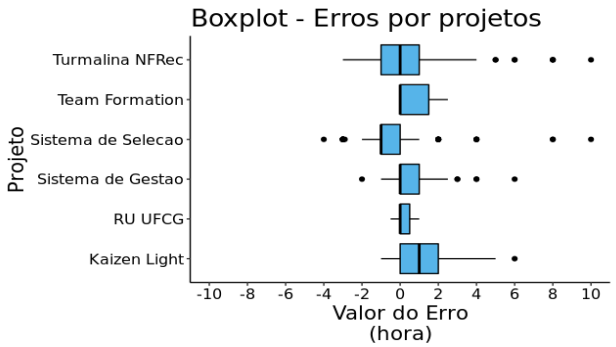

Figura 5.1 - Diagrama de Caixa do Erro x Projeto

Podemos observar que, diferente de quando comparados por categoria, os erros apresentam distribuições com alta variação quando comparados por projeto.

Tais resultados nos mostram que os erros nas estimativas possuem relação com o projeto em questão.

Para descobrir quão forte é tal relação construímos um modelo de regressão linear múltipla da seguinte forma:

$$E = aP1 + bP2 + ... + mP9,$$

onde variável dependente *E* representa o erro associado a uma tarefa e as variáveis independentes *P1, P2 ... P9* representam os projetos analisados com no mínimo 30 tarefas observadas de forma que seus valores na equação são:

- 1; quando a tarefa pertence ao projeto em questão.
- 0; quando a tarefa pertence a outro projeto.

O objetivo do modelo de regressão linear múltipla é encontrar os melhores parâmetros *a, b, ... m* de forma a minimizar a diferença entre o valor real do erro e o valor estimado pela equação.

A tabela 5.1 exibe alguns resultados obtidos da equação.

Métrica	**Valor**
Desvio Padrão do Erro	1,692
R^2	0,1195
R^2 Ajustado	0,1056

Tabela 5.1 - Resultados da regressão linear múltipla

O coeficiente de determinação (R^2) é a medida de ajustamento na regressão linear e indica, em percentagem, o quão bem o modelo consegue explicar os dados.

Para o nosso modelo, obtivemos um valor de 0,1195 para o R^2, isso nos mostra que, mesmo considerando os parâmetros que minimizam o erro da equação, ela explica apenas 11,95 % dos dados observados.

Em outras palavras, apesar da diferença de projeto impactar no erro das estimativas, tal diferença por si não é capaz de explicar totalmente os erros de estimativas nos projetos, sendo capaz de explicar apenas uma pequena porção dos dados.

6. Análise dos *outliers*

Nos diagramas de caixa vistos até o momento podemos observar a presença de valores que apresentavam um grande afastamento dos demais dados. Na estatística, esses valores recebem o nome de *outliers*. Neste tópico, analisamos tais valores na

tentativa de identificar as possíveis causas de suas discrepâncias.

Um fator que chamou atenção nos resultados analisados foi a repetição de responsáveis pelas tarefas com valores discrepantes, tanto nos dados referentes a erros positivos como nos de erros negativos. As tarefas filtradas possuíam 44 responsáveis distintos, porém, apenas 4 deles foram responsáveis pelas tarefas discrepantes. As tabelas 6.1 e 6.2 mostram respectivamente as cinco tarefas com maiores erros negativos e as cinco tarefas com maiores erros positivos. Foram utilizadas as letras X, Y, Z e K para representar os 4 responsáveis em questão.

Projeto	**Respon-sável**	**Tarefa**	**Erro**
Sistema de Seleção	Z	Criar rotas para remoção de dados	- 5
Sistema de Seleção	X	Criar rota de inserção de dados	- 5
Sistema de Seleção	X	Criar ordenação para listagem	- 3
Sistema de Seleção	K	Recuperar dados do banco de dados	- 3
Sistema de Seleção	K	Criar tela de listagem	- 2

Tabela 6.1 – Tarefas individuais com maiores erros negativos

Projeto	**Respon-sável**	**Tarefa**	**Erro**
Turmalina NFRec	X	Modificar tela de inserção de dados	20
Turmalina NFRec	X	Modificar tela de inserção de dados	14
Turmalina NFRec	X	Recuperar dados do banco de dados	12
Turmalina NFRec	Y	Modificar tela de inserção de dados	10
Sistema de Seleção	Y	Expandir informações de item da listagem	10

Tabela 6.2 – Tarefas individuais com maiores erros positivos

Note que o programador *'X'* é o responsável por cinco das dez tarefas mais discrepantes encontradas nos dados e os programadores 'Y' e 'K' são responsáveis por duas cada um. Esse fato nos alerta sobre os problemas individuais de cada programador. Mesmo considerando desenvolvedores no mesmo nível existem discrepâncias entre o nível de conhecimento de cada um.

Os diversos fatores que podem ter levado *'X'* a figurar na maior parte das tarefas com erros discrepantes vão desde uma possível dificuldade na realização de trabalhos com *interface de usuário* a problemas pessoais que possam interferir na sua produtividade. O importante aqui é atentar para o fato de que fatores individuais podem impactar as estimativas e o andamento de um projeto como um todo.

7. Conclusões e trabalhos futuros

Na nossa análise podemos concluir que alguns tipos de tarefas acumulavam maiores erros de estimativas, como por exemplo tarefas que demandam algum tipo de refatoramento de código ou validação de dados. É necessário, estudar maneiras de mitigar tais erros cometidos nesses tipos de tarefa.

Outro fator analisado foi a relação entre o projeto realizado e o erro de estimativa cometido pelos desenvolvedores iniciantes. Descobrimos que existe uma relação entre esses fatores, mas que apenas o projeto como parâmetro não é suficiente para explicar completamente os erros cometidos nas estimativas. Acreditamos que se faz necessário avaliar como cada um dos diferentes parâmetros envolvidos em cada projeto podem impactar nas estimativas realizadas bem como estudar maneiras de como reduzir esses impactos.

Por fim, alertamos para os problemas individuais de cada profissional, sejam eles causados por falta de conhecimento em uma área específica ou mesmo por problemas pessoais. É importante que possamos perceber a ocorrência desses problemas o quanto antes para evitar que eles prejudiquem o andamento do projeto como um todo. Recomendamos, para esse caso, a utilização de técnicas baseadas na análise de dados dos projetos, semelhantes às utilizadas nesse estudo, aplicadas com frequência durante todo o ciclo de desenvolvimento.

REFERÊNCIAS

[1] Emanuel Dantas (2018), Effort Estimation in Agile Software Development: an Updated Review, International Journal of Software Engineering and Knowledge Engineering.

[2] George Stark (2011), A Comparison of Parametric Software Estimation Models Using Real Project Data, IBM Global Services

[3] S. M. Sutton, Jr. (2018), Informed Projection: Using What You Know to Make Simple Estimates of Work Better.

[4] Emanuel Dantas (2019), An Effort Estimation Support Tool for Agile Software Development: An Empirical Evaluation.

[5] Barry Boehm (2006), Minimizing Future Guesswork in Estimating, IBM Conference on Atlanta, Ga. Feb. 2006.

[6] F. Arcelli Fontana (2015), On experimenting refactoring tools to remove code smells, XP '15 workshops Scientific Workshop Proceedings of the XP2015, Article No. 7.