An Evaluation of Influence Human Personality in Software Development: an experience report

Anderson S. Barroso
Universidade Federal de Sergipe
Sergipe, Brasil
giga.anderson@gmail.com

Jamille S. Madureira
Universidade Federal de Sergipe
Instituto Federal de Sergipe
Sergipe, Brasil
jamillemadureira@gmail.com

Fabrício S. Melo
Universidade Federal de Sergipe
Sergipe, Brasil
fs-melo@bol.com.br

Thiago D. S. Souza
Universidade Federal de Sergipe
Sergipe, Brasil
thiago.darley@gmail.com

Michel dos S. Soares
Universidade Federal de Sergipe
Sergipe, Brasil
mics.soares@gmail.com

Rogerio P. C. do Nascimento
Universidade Federal de Sergipe
Sergipe, Brasil
rogerio@ufs.br

ABSTRACT

Analyze software developers personality has been a topic discussed by many researchers and, over the past few years, has applied different theoretical models of psychology with the intention of promoting a motivation that make the most effective teams. Thus, we intend to improve the performance of programmers and enable better orchestration software development team. This research conducted a case study where was done a study of the personality of a group of software developers of a particular educational institution, through the Myers-Briggs Type Indicator model, to associate the results achieved to object-oriented metrics software guided the source code they developed. As result, we can not say that the three personalitie types identified, significantly influenced the quality of the built software. However an developer type showed significantly better characteristics with respect to line-of-code metric.

Keywords

Personality; object-oriented software metrics; software development; MBTI

1. INTRODUÇÃO

As organizações modernas estão cada vez mais concentrando seus esforços em criatividade e inovação, enquanto se empenham para se manterem competitivas em relação às mudanças produzidas pelo mercado. Essa atitude influencia diretamente a área de TIC (Tecnologia da Informação e Comunicação), pois ela precisa estar totalmente alinhada ao Planejamento estratégico da empresa [19]. Para isso, metodologias, técnicas e tecnologias de desenvolvimento de softwares vem sendo criadas e aprimoradas nos últimos anos, incentivando constantes atualizações e treinamentos para a equipe de desenvolvimento de software [14]. É importante saber em qual atividade o desenvolvedor terá melhor rendimento. Para isso surge a necessidade de identificar os traços de personalidade de um desenvolvedor e tentar abstrair, desta análise, padrões que indiquem um caminho para uma melhor orquestração da equipe de desenvolvimento.

Historicamente, pesquisadores tem se preocupado com a personalidade de desenvolvedores de softwares, como mostram as pesquisas feitas por Bartol et al [6] em 1982, Pocius em 1991 [28], Cruz et al em 2011 [9] e Varona em 2012 [35]. Na maioria das pesquisas são mostradas tentativas de identificar a relação entre padrões emocionais, motivacionais e comportamentais e o ambiente de desenvolvimento de softwares, com o intuito de analisar qual desenvolvedor se encaixa melhor em uma determinada tarefa, possibilitando a criação de equipes mais eficazes [9].

Neste trabalho, foi feita uma análise em repositórios de softwares orientado a objetos de uma empresa privada para tentar associar a personalidade de um desenvolvedor de software aos códigos-fontes que ele desenvolveu por meio das métricas de software do código desenvolvido. O objetivo deste trabalho foi verificar se a personalidade do desenvolvedor tem influência na qualidade do software produzido.

Para isso, a personalidade dos desenvolvedores foi analisada utilizando o modelo MBTI [22] e seus respectivos códigos-fonte foram avaliados por meio de métricas de software. Finalmente, essas informações foram analisadas para verificar a existência de padrões que permitam identificar a influência da personalidade do desenvolvedor nos softwares por ele desenvolvidos.

Este artigo está dividido em seis partes, a saber: na seção 2 são apresentados os trabalhos relacionados, na seção 3 é descrito o referencial teórico sobre personalidade de software e métricas de softwares orientados a objetos, na seção 4 é retratada a metodologia adotada para a realização da pesquisa, na seção 5 são demonstrados os resultados e, por fim,

EATIS '16 Cartagena das Índias, Bolívar, Colômbia

DOI:

na seção 6 é apresentada a conclusão e os trabalhos futuros.

2. TRABALHOS RELACIONADOS

Ao longo do tempo, vários estudos foram feitos com o intuito de associar a personalidade às atividades exercidas no âmbito da Engenharia de Software. A aplicação de métricas de software é reconhecida como fator importante para o acompanhamento do projeto e análise da qualidade do produto desenvolvido.

Em [33], os autores realizaram uma avaliação empírica da influência da personalidade humana na área de testes de software. Foi desenvolvido um teste chamado ETAT (*Exploratory Testing Aptitude Test*) e os resultados mostraram que há uma relação positiva entre o teste aplicado e a personalidade humana, no qual foi evidenciado que o tipo de personalidade extrovertida caracteriza indivíduos mais suscetíveis a serem bons testadores. Como uma limitação desse estudo e ameaça à validade, os autores retratam que o trabalho foi delimitado a uma determinada localização geográfica, e para os alunos de engenharia de software, devido ao pouco tempo e recursos.

Como indicativo de que a personalidade influencia nas atividades dos desenvolvedores de softwares, Wiesche et al [37] fizeram uma análise do tipo MBTI [22] e concluíram que há características conhecidas como introvertidas e extrovertidas, originalmente definidas por Jung, que podem ajudar a definir a melhor personalidade para as tarefas propostas para um desenvolvedor. Em sua pesquisa, eles descobriram que as tarefas que exigem desenvolvimento individual e criatividade combinam com pessoas de personalidade introvertida, enquanto as tarefas que exigem colaboração, como programação em pares, combinam com pessoas de personalidade extrovertida. Os autores retratam que o tema requer uma discussão mais aprofundada, pois existem modelos psicológicos adjacentes que são aplicados em um contexto particular e que há tarefas de engenharia de softwares não debatidas no estudo.

A pesquisa descrita em [36] explica que a necessidade de medição para permitir um controle eficaz em projetos de software é bem reconhecida. É preciso compreender a progressão técnica de projetos, especialmente onde os estes têm altos níveis de complexidade. As investigações evidenciam a dificuldade tanto na implantação e utilização de um regime de medição, quanto na divulgação dos resultados para a organização. A administração das medições é útil para as organizações, pois ajuda a controlar o gerenciamento de funções. Na pesquisa é apresentada uma proposta para a criação de uma ferramenta KM (*Knowledge Management*) que permitirá o controle dos projetos, coletando, armazenando e exibindo as informações para o público desejado.

Baseado nos resultados satisfatórios ao aplicar modelos psicológicos encontrados nos trabalhos de [33] e [37] e levando em consideração a importância de se medir a qualidade de um software, encontrado em [36], foi feito, neste trabalho, um estudo para identificar se existe relação entre a personalidade do desenvolvedor e a qualidade do software por ele produzido.

3. REFERENCIAL TEÓRICO

3.1 MBTI - Myers-Briggs Type Indicator

É muito comum na análise de personalidade de profissionais de TI utilizar o teste baseado no modelo MBTI, que pode resultar em 16 tipos de personalidades diferentes, como mostrado na tabela 1.

O modelo MBTI é o mais aplicado para identificar a personalidade de desenvolvedores de software e é baseado no modelo de Carl Jung [22][23].

Divide a personalidade humana em três dimensões:

- Como as pessoas se relacionam com o mundo;
- A forma como a informação é conhecida e;
- A forma como a informação é processada.

Sendo assim, o modelo identifica quatro pares de preferências conhecidos como dicotomias. São eles [22][23]:

- Atitudes (E-I):
  - Introversão(I): Costumam estar envolvidos com ideias; preferem refletir antes de agir. Precisam de tempo para pensar e recuperar sua energia. Em geral, são pouco sociáveis.
  - Extroversão(E): Costumam agir; gostam de realizar várias atividades; agem primeiro e depois pensam. Quando inativos, sua energia diminui. Em geral, são sociáveis.
- Funções (S-N e T-F):
  - Sensoriais(S): Confiam mais em coisas palpáveis, concretas, informações sensoriais. Gostam de detalhes e fatos. Para eles o significado está nos dados. Precisam de muitas informações;
  - Intuitivas(N): Preferem informações abstratas e teóricas, que podem ser associadas com outras informações. Gostam de interpretar os dados com base em conhecimento prévio. Trabalham bem com informações incompletas e dedutíveis.
  - Racionalistas(T): Decidem com base na lógica e procuram argumentos racionais.
  - Sentimentais(F): Decidem com base em seus sentimentos e não com as emoções.
- Estilo de vida (J-P):
  - Julgadores(J): Sentem-se tranquilos quando as decisões são tomadas;
  - Pensadores(P): Sentem-se tranquilos deixando as opções em aberto;

A combinação desses 4 pares podem resultar em 16 tipos de personalidades diferentes. O tipo de personalidade é formado por 4 letras. A primeira letra representa a dicotomia Atitude (Introversão(I) ou Extroversão(E)), a segunda e a terceira letras representam a dicotomia Funções (Sensorial(S) ou Intuitivo(N); Pensador(T) ou Sentimental(F)) e a quarta letra representa a dicotomia Estilo de Vida (Julgador(J) ou Perceptivo(P)) [22][23].

Esse modelo foi escolhido por se tratar de um indicador que tem sido usado há mais de 50 anos para identificar o tipo de personalidade de um indivíduo e suas preferências, conforme pesquisado por Kaiser [18], Gorla et. al [16] e Sach et. al [32] em seus respectivos trabalhos.

Table 1: Tipos de Personalidade

	Sensorial (S)	**Sensorial (S)**	**Intuitiva (N)**	**Intuitiva (N)**	
Introversão (I)	ISTJ (Inspetor)	ISFJ (Protetor)	INFJ (Conselheiro)	INTJ (Mentor)	**Julgador (J)**
Introversão (I)	ISTP (Artesão)	ISFP (Compositor)	INFP (Idealista)	INTP (Arquiteto)	**Pensador (P)**
Extroversão (E)	ESTP (Dínamo)	ESFP (Animador)	ENFP (Campeão)	ENTP (Visionário)	**Pensador (P)**
Extroversão (E)	ESTJ (Supervisor)	ESFJ (Provedor)	ENFJ (Professor)	ENTJ (Comandante)	**Julgador (J)**
	Racionalista (T)	**Sentimental (F)**	**Sentimental (F)**	**Racionalista (T)**	

3.2 Métricas de Software OO

A Engenharia de Software possui diversos tipos de métricas que tem sido aplicadas na medição, tanto do processo quanto do produto de software. Dentre aquelas que avaliam o produto, pode-se citar métricas para o modelo de requisitos, para o código-fonte e também para o modelo do projeto. Neste trabalho, o foco são as métricas para projeto orientado a objetos.

Para esse estudo foram selecionadas métricas para elementos do paradigma orientado a objetos, por serem frequentemente utilizadas por pesquisadores na Engenharia de Software [29]. Foram utilizadas na área de redução de falhas [13], em características como manutenibilidade, testabilidade e compreensibilidade [25], na manutenção de softwares orientados a objeto livres [17] e na previsão de refatoração [5].

Outro ponto positivo é que, de acordo Radjenovi et al [29], em uma revisão sistemática realizada, os autores identificaram que métricas orientadas a objetos foram usadas aproximadamente duas vezes mais (49%) que as métricas tradicionais (27%) e métricas de processo (24%). Eles afirmam ainda que o conjunto de métricas CK (Chidamber e Kemerer) [8] é o mais popular entre as métricas orientadas a objetos, por serem bem difundidas no meio acadêmico e frequentemente utilizadas em estudos.

As métricas CK [8] foram definidas por Chidamber e Kemerer em 1994. O objetivo das métricas é medir a complexidade do projeto em relação ao seu impacto sobre atributos de qualidade como usabilidade, facilidade de manutenção, funcionalidade e confiabilidade. Seis métricas foram definidas: *Weighted Methods Per Class* (WMC), *Depth of Inheritance Tree* (DIT), *Number of Children* (NC), *Coupling between Object Classes* (CBO), *Response For a Class* (RFC) e *Lack of Cohesion in Methods* (LCM) [11].

Além das métricas CBO e DIT do conjunto CK, foram utilizadas as métricas Complexidade Ciclomática (CC) , Linhas de Código (LOC) e Índice de Manutenibilidade (MI). A complexidade ciclomática [20] calcula a quantidade de caminhos independentes a partir do código fonte. A métrica Linhas de Código conta cada linha de código física em um programa, excluindo as linhas em branco e comentários. Apesar de alguns autores terem apontado as deficiências da métrica LOC, esta ainda é usada em prática devido à sua simplicidade [38]. O índice de manutenibilidade [26] determina o grau de manutenibilidade do software baseado no status do respectivo código-fonte.

4. METODOLOGIA

O objetivo desse trabalho foi avaliar o fator humano em desenvolvimento de software. Desta forma, realizou-se um estudo de caso em uma instituição particular de ensino superior, onde foram avaliadas as personalidades de seis desenvolvedores e qualidade do software desenvolvido por eles.

A experiência dos programadores em desenvolvimento de software é mostrada na tabela 2. Como pode ser observado, os programadores possuem escolaridades e experiências variadas:

Table 2: Participantes X Experiência

Programador	Escolaridade	Experiência com programação OO (anos)
1	Superior	6
2	Pós-Graduação	7
3	Pós-Graduação	8
4	Superior	8
5	Superior	6
6	Superior	5

Para traçar a personalidade dos desenvolvedores de software, foi feito um teste psicológico baseado no modelo MBTI de Myers-Briggs [23]. Um grupo de seis desenvolvedores realizou o teste psicológico utilizando a ferramenta 16 Personalities [3].

As métricas foram colhidas por meio da ferramenta CodeAnalysis [1], parte integrante do Visual Studio 2010, da Microsoft, que é o ambiente de desenvolvimento orientado a objeto utilizado pela instituição escolhida. O código armazenado encontrava-se no servidor TFS(*Team Foundation Server*), que é uma plataforma de colaboração de gerenciamento do ciclo de vida de aplicativos da Microsoft [4].

Como optamos por obter as métricas de software que utilizam o paradigma de orientação a objetos, a empresa escolhida, uma instituição de ensino privado, foi identificada como detentora de todos os pré-requisitos necessários para a execução dos teste: possuía uma estrutura de desenvolvimento bem definida, utilizando o Visual Studio da Microsoft e controlava as alterações realizadas nos software em um controle de versionamento.

Com base nesse ambiente, identificamos 6 sistemas que foram desenvolvidos apenas por um mesmo desenvolvedor. O número reduzido de participantes se justifica pela dificuldade de se encontrar, na nossa região, sistemas que somente um desenvolvedor tenha desenvolvido. O local que encontramos mais casos foi na instituição escolhida.

Escolhidos os programadores, para cada um foi analisado o projeto desenvolvido somente por ele. Os sitemas possuíam tamanho e complexidade variados. Após a aplicação dos testes, as métricas DIT, CC, CBO e MI de cada desenvolvedor foram coletadas de softwares produzidos em C# e .NET usando a ferramenta automatizada CodeAnalysis do Microsoft Visual Studio 2010 e gerados dados para mineração.

Após a aplicação das métricas e testes psicológicos, os resultados foram analisados para verificar a existência de padrões que permitam identificar a influência da personalidade do desenvolvedor nos software por ele desenvolvidos. A relação entre o índice de manutenibilidade (MI) e várias

métricas em estudo foi identificada com base na análise de correlação de Pearson [27].

Foi realizado o teste de Friedman [15] para verificar a existência de diferenças significativas entre as métricas de software de cada tipo de personalidade, adotando um nível de significância de 5%. A principal vantagem deste teste é que ele pode atuar em amostras que não obedecem necessariamente à condição de normalidade. Assim, é suficientemente adequado e apresenta um excelente desempenho para efetuar comparações múltiplas em múltiplos problemas [10] e tem sua representatividade reconhecida em ciência da computação principalmente em avaliação de algoritmos de otimização combinatória [10] [12] [24].

5. RESULTADOS

Os seis desenvolvedores realizaram um teste psicológico baseado no modelo MBTI por meio da ferramenta 16 Personalities [3]. Dos dezesseis tipos de personalidades diferentes que o modelo MBTI classifica, foram identificados quatro tipos no estudo.

Os tipos identificados pelo modelo MBTI possuem características específicas de personalidade que descrevem o comportamento de um indivíduo e sugere como este pode reagir em situações específicas. Neste estudo, foram identificados dois programadores do tipo ESTJ, dois programadores do tipo ESFJ, um programador do tipo ENFP e um programador do tipo INTJ, como é mostrado na tabela 3.

Table 3: Relação programador X personalidade

Tipo	Programador	Perfil
ESTJ	1,3	Administradores excelentes, inigualáveis em gerir coisas ou pessoas
ESFJ	2,6	Pessoas muito atenciosas, sociais e populares,sempre prontas para ajudar;
INTJ	4	Pensadores criativos e estratégicos, com um plano para tudo.
ENFP	5	Possuem espíritos livres, criativos, sociáveis que encontram sempre uma razão para estar feliz

Os dados sobre dos tipos de personalidade identificados nos seis programadores foram distribuídos conforme a tabela 4.

As figuras de 1 a 5 apresentam as distribuições de frequência para cada métrica comparando os tipos de personalidade identificados. O eixo X representa os valores da métrica. O eixo Y representa a frequência relativa de classes encontradas em cada nível das métricas CBO e DIT. Enquanto que o eixo Y das métricas CC, LOC e MI representa a frequência relativa de métodos de classes encontrados em cada um de seus níveis.

A maioria dos métodos dos softwares analisados, originaram classes com maior predominância de alto acoplamento (CBO $>$ 0) (Figura 1) e grande profundidade de herança (DIT $\leq$ 3) (Figura 3) [8], no entanto, no que se refere à tamanho e complexidade de software, nota-se que as métricas MI e LOC indicam uma boa qualidade de desenvolvimento, pois todo método de software analisado foi desenvolvido com um índice de manutenibilidade sustentável (entre 20 e 100) [2], reflexo da maior predominância de métodos com até 10 linhas de códigos (Figura 4).

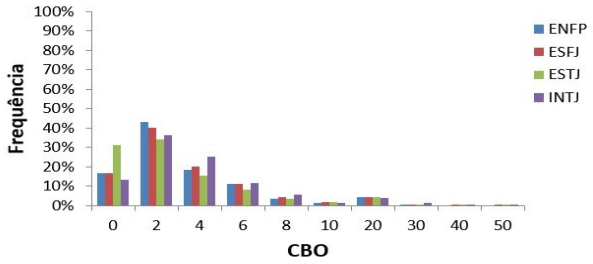

Figure 1: Distribuição de frequência da métrica CBO por tipo de personalidade

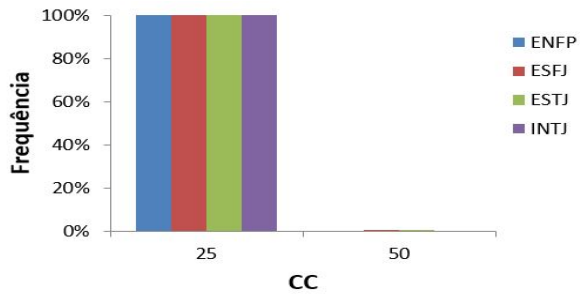

Figure 2: Distribuição de frequência da métrica CC por tipo de personalidade

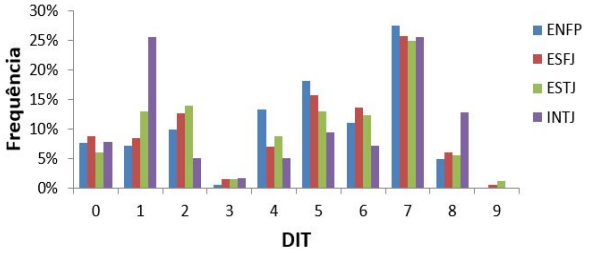

Figure 3: Distribuição de frequência da métrica DIT por tipo de personalidade

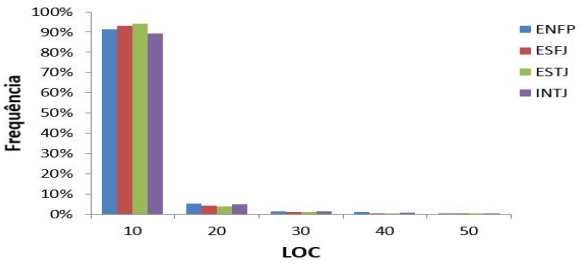

Figure 4: Distribuição de frequência da métrica LOC por tipo de personalidade

Table 4: Resultados dos testes de personalidade

Programador	Tipo MBTI	Percentuais				
1	ESTJ	18 % Extrovertido	21% Observador	18% Pensante	13% Julgador	14% Assertivo
2	ESFJ	18 % Extrovertido	45% Observador	57% Sentimental	14% Julgador	29% Assertivo
3	ESTJ	18 % Extrovertido	32% Observador	10% Pensante	35% Julgador	30% Assertivo
4	INTJ	12 % Introvertido	13% Intuitivo	50% Pensante	52% Julgador	66% Assertivo
5	ENFP	28 % Extrovertido	8% Intuitivo	32% Sentimental	9% Explorador	21% Cauteloso
6	ESFJ	52 % Extrovertido	14% Intuitivo	16% Pensante	20% Julgador	47% Assertivo

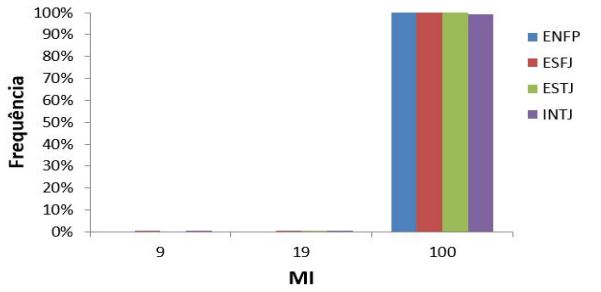

Figure 5: Distribuição de frequência da métrica MI por tipo de personalidade

Foi observado que, na maioria da produção dos softwares analisados, foram originadas classes de baixo acoplamento e profundidade de herança menor que 3, estando dentro dos padrões encontrados na literatura.

Na tabela 5 o valor de $p < 0,05$ indica que existe uma diferença significativa na métrica LOC entre os tipos de personalidades encontrados.

Table 5: Tipo de personalidade x LOC

Diferenças	valor-p
ESFJ - ENFP	0,9930
ESTJ - ENFP	0,1296
INTJ - ENFP	0,1295
ESTJ - ESFJ	0,0683
INTJ - ESFJ	0,2257
INTJ - ESTJ	0,0001

Nota-se então que, embora de diferentes tipos, dentre os desenvolvedores analisados aqueles de tipo ENFP, ESFJ e INTJ produziram resultados com igual qualidade a uma significância de 5%. No entanto foi notada uma diferença significante entre o par INTJ e ESTJ. Bem como uma diferença parcialmente significante entre o par ESTJ e ESFJ. Uma considerável diferença entre o tipo INTJ e os demais é notada no uso de herança. Enquanto os demais tipos implementam softwares com predominância de DIT=7, o tipo INTJ, na maioria de suas implementações, usou DIT=1, reconhecida como uma excelente pratica de projeto de software. O que corrobora com o modelo de Myers [21], ao afirmar que pessoas desse tipo são solucionadores de problemas analíticos e apresentam alta capacidade de raciocínio lógico e solução de problemas complexos.

Com base nos coeficientes de correlação apresentados na tabela 6 foi observada uma forte correlação entre o índice de manutenibilidade (MI) e o acoplamento (CBO). Por outro lado observou-se que, apesar de serem parte direta do cálculo do MI, as métricas CC e LOC apresentaram uma moderada correlação com esse índice [30].

Table 6: Análise de Correlação

Fonte da variação	Correlação com índice de manutenibilidade (MI)
CC	-0,56
LOC	-0,45
CBO	-0,70

Em todos os casos observa-se que o incremento de complexidade, de linhas de código e de acoplamento diminui a manutenibilidade do software.

6. CONCLUSÃO

Nesse contexto, sob diferentes perspectivas de análise e levando em consideração uma amostra de seis desenvolvedores de softwares com experiências variadas, não se pode afirmar que as personalidades ENFP, ESFJ e ESTJ influenciam diretamente na qualidade do software construído. No entanto o desenvolvedor de tipo INTJ apresentou características significativamente diferenciadas em relação a qualidade de software, mantendo menores níveis de profundidade de herança e métodos de menor tamanho.

Porém, vale ressaltar que este estudo foi aplicado apenas em uma localização geográfica com programadores (indústria) que possuem uma cultura própria de desenvolvimento de software e, para as conclusões definitivas, será preciso um estudo mais abrangente sobre a relação personalidade-métricas OO.

Como trabalho futuro, sugere-se a realização da mesma análise por meio de um experimento controlado, no qual desenvolvedores implementem a mesma tarefa e as métricas colhidas desta implementação sejam associadas às suas respectivas personalidades.

7. AGRADECIMENTOS

Agradecemos ao programa de Mestrado em Ciência da Computação (PROCC) da Universidade Federal de Sergipe - Brasil.

8. REFERENCES

[1] Code analysis tools. https: msdn.microsoft.com/en-us/library/dd264897.aspx. Accessed:04/11/2015.

[2] Measuring complexity and maintainability of managed code. https: //msdn.microsoft.com/en-us/library/bb385910.aspx. Accessed: 2015-11-04.

[3] Neris analytics, free personality test. http:www.16personalities.com/free-personality-test. Accessed: 10/20/2015,.

[4] Team foundation server. https: msdn.microsoft.com/pt-br/vstudio/ff637362.aspx. Accessed:05/11/2015,.

[5] J. Al Dallal. Constructing models for predicting extract subclass refactoring opportunities using object-oriented quality metrics. *Information and Software Technology*, 54(10):1125–1141, 2012.

[6] K. M. Bartol and D. C. Martin. Managing information systems personnel: a review of the literature and managerial implications. *MIS Quarterly*, pages 49–70, 1982.

[7] S. J. Blatt. Where have we been and where are we going? reflections on 50 years of personality assessment. *Journal of personality assessment*, 50(3):343–346, 1986.

[8] S. R. Chidamber and C. F. Kemerer. A metrics suite for object oriented design. *IEEE Transactions on Software Engineering*, 20(6):476–493, 1994.

[9] S. S. Cruz, F. Q. da Silva, C. V. Monteiro, C. Santos, and M. Dos Santos. Personality in software engineering: Preliminary findings from a systematic literature review. In *15th Annual Conference on Evaluation & Assessment in Software Engineering (EASE 2011)*, pages 1–10. IET, 2011.

[10] J. Derrac, S. García, D. Molina, and F. Herrera. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1):3–18, 2011.

[11] S. K. Dubey and A. Rana. A comprehensive assessment of object-oriented software systems using metrics approach. *International Journal on Computer Science and Engineering*, 2(08):2726–2730, 2010.

[12] D. Elmazi, T. Oda, S. Sakamoto, E. Spaho, L. Barolli, and F. Xhafa. Friedman test for analysing wmns: A comparison study for genetic algorithms and simulated annealing. In *9th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2015*, pages 171–178. IEEE, 2015.

[13] N. Fenton and J. Bieman. *Software metrics: a rigorous and practical approach*. CRC Press, 2014.

[14] V. Ferreira, N. Natasha, and J. J. Langerman. The correlation between personality type and individual performance on an ICT Project. In *9th International Conference on Computer Science & Education (ICCSE), 2014*, pages 425–430. IEEE, 2014.

[15] M. Friedman. A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, 11(1):86–92, 1940.

[16] N. Gorla and Y. W. Lam. Who should work with whom?: building effective software project teams. *Communications of the ACM*, 47(6):79–82, 2004.

[17] K. Johari and A. Kaur. Validation of object oriented metrics using open source software system: an empirical study. *ACM SIGSOFT Software Engineering Notes*, 37(1):1–4, 2012.

[18] K. M. Kaiser and R. P. Bostrom. Personality characteristics of mis project teams: An empirical study and action-research design. *MIS Quarterly*, pages 43–60, 1982.

[19] S. Luo, H. Chai, and L. Liu. Based on the balanced scorecard: Performance evaluation of knowledge team. In *2012 International Conference on Information Management, Innovation Management and Industrial Engineering*, volume 3, pages 352–355, 2012.

[20] T. J. McCabe. A complexity measure. *IEEE Transactions on Software Engineering*, (4):308–320, 1976.

[21] I. B. Myers. The myers-briggs type indicator: Manual (1962). 1962.

[22] I. B. Myers, M. H. McCaulley, and R. Most. *Manual, a guide to the development and use of the Myers-Briggs type indicator*. Consulting Psychologists Press, 1985.

[23] I. B. Myers, M. H. McCaulley, N. L. Quenk, and A. L. Hammer. *MBTI manual: A guide to the development and use of the Myers-Briggs Type Indicator*, volume 3. Consulting Psychologists Press Palo Alto, CA, 1998.

[24] T. Oda, Y. Liu, S. Sakamoto, D. Elmazi, L. Barolli, and F. Xhafa. Analysis of mesh router placement in wireless mesh networks using friedman test considering different meta-heuristics. *International Journal of Communication Networks and Distributed Systems*, 15(1):84–106, 2015.

[25] S. Olbrich, D. S. Cruzes, V. Basili, and N. Zazworka. The evolution and impact of code smells: A case study of two open source systems. In *Proceedings of the 2009 3rd international symposium on empirical software engineering and measurement*, pages 390–400. IEEE Computer Society, 2009.

[26] P. Oman and J. Hagemeister. Metrics for assessing a software system's maintainability. In *Software Maintenance, 1992. Proceerdings., Conference on*, pages 337–344. IEEE, 1992.

[27] K. Pearson. Mathematical contributions to the theory of evolution.–on a form of spurious correlation which may arise when indices are used in the measurement of organs. *Proceedings of the royal society of london*, 60(359-367):489–498, 1896.

[28] K. E. Pocius. Personality factors in human-computer interaction: A review of the literature. *Computers in Human Behavior*, 7(3):103–135, 1991.

[29] D. Radjenović, M. Heričko, R. Torkar, and A. Živkovič. Software fault prediction metrics: A systematic literature review. *Information and Software Technology*, 55(8):1397–1418, 2013.

[30] D. J. Rumsey. *Statistics for dummies*. John Wiley & Sons, 2011.

[31] R. H. Rutherfoord. Using personality inventories to form teams for class projects: a case study. In *Proceedings of the 7th Conference on Information technology education*, pages 9–14. ACM, 2006.

[32] R. Sach, M. Petre, and H. Sharp. The use of mbti in software engineering. 2010.

[33] L. Shoaib, A. Nadeem, and A. Akbar. An empirical evaluation of the influence of human personality on exploratory software testing. In *IEEE 13th International Multitopic Conference, 2009. INMIC 2009*, pages 1–6. IEEE, 2009.

[34] S. Srivastava and R. Kumar. Indirect method to

measure software quality using CK-OO suite. In *International Conference on Intelligent Systems and Signal Processing (ISSP)*, pages 47–51. IEEE, 2013.

[35] D. Varona, L. F. Capretz, Y. Piñero, and A. Raza. Evolution of software engineers' personality profile. *ACM SIGSOFT Software Engineering Notes*, 37(1):1–5, 2012.

[36] T. Walworth, M. Yearworth, and L. Shrieves. Knowledge management for metrics: Enabling analysis and dissemination of metrics. In *8th Annual IEEE Systems Conference (SysCon), 2014*, pages 199–205. IEEE, 2014.

[37] M. Wiesche and H. Krcmar. The relationship of personality models and development tasks in software engineering. In *Proceedings of the 52nd ACM conference on Computers and people research*, pages 149–161. ACM, 2014.

[38] H. Zhang. An investigation of the relationships between lines of code and defects. In *IEEE International Conference on Software Maintenance, 2009. ICSM 2009*, pages 274–283. IEEE, 2009.