ANÁLISE DE PREFERÊNCIAS DE AMBIENTES DE DESENVOLVIMENTO INTEGRADOS NO MEIO ACADÊMICO

MAIRON SCHNEIDER CARDOSO[1]; MATEUS MOREIRA SILVEIRA DO NASCIMENTO[2]

[1]Universidade Federal de Pelotas - maironcardoso@inf.ufpel.edu.br
[2]Universidade Federal de Pelotas – mmsdnascimento@inf.ufpel.edu.br

1. INTRODUÇÃO

Com o rápido avanço tecnológico nas áreas computacionais na década de 80, o método arcaico da programação, dos anos 60, em simples editores de texto (KLEINA et al., 2011) tornou-se cada vez mais obsoleto, pois com o aumento dos códigos, proveniente do avanço, impossibilitou a detecção de erros de maneira rápida. Em 1983 a empresa **Borland Ltd** investiu no mercado computacional, apostando na linguagem Pascal, a empresa adquiriu um editor e compilador do dinamarquês **Anders Hejlsberg**. O usufruto da nova tecnologia viabilizou a detecção mais rápida de erros, além de ser muito prático, como conta o desenvolvedor da Microsoft na época **Charles Petzold** (PATRIZIO et al., 2013)**.**

Depois de alguns anos, em 1990, com o surgimento do Windows 3.0 e o seu ambiente operacional, que ilustrava a noção de um "desktop gráfico", **Microsoft's** lança seu primeiro **Ambiente de desenvolvimento integrado** ou **Integrated Development Environment (IDE**) o termo em inglês, popularizando a sigla entre os profissionais da área. A primeira IDE era chamada de **Visual Studio,** muito popular entre os desenvolvedores da época (Microsoft, 1997), visto que, descomplicava algoritmos extensos com suas cores, útil para evidenciar funções diferentes, permitia ao usuário a compilação dentro do próprio software, fundando um cenário ideal para a programação de códigos extensos, pois facilitava a visualização dos códigos e possuía também uma rápida detecção de erros.

A ideia de um editor de texto, customizável, que se relacione em conjunto com o compilador nos acompanha até hoje, possuindo uma enorme competição entre os desenvolvedores desses softwares, entretanto, a tecnologia avançou e com ela as necessidades de novas funções e diferenciais foram geradas. Isto é, os critérios de avaliação dos principais **IDEs** multiplicaram-se, gerando diferentes possibilidades de escolhas.

2. METODOLOGIA

Foram realizados uma coleta de informações, aplicado a 55 alunos de Ciência da computação e Engenharia de computação que cursam a graduação ou pós-graduação da Universidade Federal de Pelotas, um questionário que solicitava as informações sobre preferências de ambientes de desenvolvimento integrado. Utilizando as informações coletadas foi possível, através de várias tabelas, avaliar a preferência dos alunos, observando também as possíveis vantagens e desvantagens de certas IDEs e gerando a possibilidade do estudo das preferências dos alunos em questão.

2.1 IDEs utilizadas

A escolha de IDEs para anexação na pesquisa foi realizada com base nas IDEs mais utilizadas pelos alunos iniciantes e avançados na área da programação, explicitando a diferença de IDEs mais simples (Geany, Sublime text), com as profissionais (Visual Studio, Eclipse). Tal diferença é explicada por Luiz F. Noschang (2014), utilizando a linguagem dos programas (inglês) como uma das culpadas pela difícil utilização por alunos iniciantes, além disso, segundo ele a interpretação dos erros se torna mais difícil em IDEs com níveis mais complexos.

3. RESULTADOS E DISCUSSÃO

A Tabela 1 demonstra a pesquisa relacionada a preferências de Ambientes de desenvolvimento integrado (IDE) entre os alunos, ilustrando a quantidade de alunos (QA) e a porcentagem de preferência (PA):

Alunos	IDE	QA	PA
55	Emacs	4	7.27%
	Geany	12	21.81%
	VIM	3	5.45%
	Sublime Text	16	29.09%
	Visual Studio	5	9.09%
	Atom	3	5.45%
	CodeBlocks	2	3.63%
	Dev C++	2	3.63%
	NetBeans	5	9.09%
	Eclipse	2	3.63%
	Xcode	1	1.81%

Tabela 1. Alunos, IDE's favoritas, Quantidade de Alunos, Porcentagem de Preferência.

Observa-se na Tabela 1 e fica claro portanto a popularidade de certos Ambientes de Desenvolvimento integrado, entretanto, na sua maioria básicos de forma que cumpram suas necessidades como programadores, entretanto, ainda que não tão comum, não exclui a possibilidade de utilização de IDEs profissionais para projetos mais elaborados.

A Tabela 2 elucida a diferença de complexidade de IDE's na resolução de problemas com diferentes níveis de dificuldade, definidos por tamanho do algoritmo (TA), usando IDES complexas (IC) ou IDE's simples (IS):

Alunos	TA	IC	IS
55	Longos	41	14
	Curtos	20	35

Tabela 2. Alunos, Tamanho Do Algorítimo, IDE Complexa, IDE Simples.

Observando a Tabela 2 é possível notar que programadores experientes ou até mesmo iniciantes, buscam conseguir dominar pelo menos dois Ambiente de desenvolvimento integrado, criando a possibilidade de alternar a estação de trabalho para resolução de problemas extensos ou curtos.

A tabela 3 expõe as funções ideais em cada IDE (FII) aplicada aos Alunos, demonstrando a Quantidade de Alunos (QA) e a Porcentagem de Preferência (PA):

Alunos	FII	QA	PA
55	Customização(Cores, indicações).	12	21.81%
	Praticidade (Em relação a execução do programa)	21	38.18%
	Suporte para diversas linguagens.	11	20%
	Desempenho nos PC's.	9	16.36%
	Debugging eficiente	2	3.63%

Tabela 3. Alunos, Funções importantes em uma IDE, Porcentagem de preferência.

Concluímos então ao observar a tabela 3 que é imprescindível a praticidade na execução do programa, aumentando a velocidade na qual os programadores conseguem resolver problemas, entretanto, notamos também que a customização é de extrema ajuda, visto que, facilita a visualização do código.

A tabela 4 explicita as opiniões dos Alunos em relação as IDEs mais recomendadas para pessoas que queiram aprender a programar, utilizando a Quantidade de Alunos (QA) e a Porcentagem de Preferência (PA):

Alunos	IDE	QA	PA
55	Geany	35	63.63%
	Sublime text	9	16.36%
	Visual Studio	5	9.09%
	Dev C++	3	5.45%
	CodeBlocks	1	1.81%
	Emacs	1	1.81%
	Atom	1	1.81%

Tabela 4. Alunos, Quantidade de Alunos, Porcentagem de preferência.

É possível notar ao observar a tabela 4 que o Geany se destaca como IDE ideal para iniciantes na área da programação, pois ao relacionar com a tabela 3 é possível destacar a excepcional importância da questão de praticidade na execução de um programa, presente na sua estrutura, combinado com uma eficiente customização das indicações, tornando-se a principal escolha para usuários inexperientes.

4. CONCLUSÕES

Poderíamos concluir que os alunos deveriam ir conhecendo progressivamente IDEs conforme suas necessidades computacionais fossem se aprofundando, aproveitando-se das funções disponibilizadas pelos Ambientes de desenvolvimento integrado de maneira progressiva. Também é importante ressaltar a importância da escolha correta da IDE como forma de agilizar o processo de escrita de um algoritmo, visto que, para códigos com um nível de complexidade baixo é desnecessário o uso de IDEs com uma quantidade exacerbada de utilitários, diminuindo a necessidade de uma alta compreensão de IDEs e gerando a possibilidade de maximização de desempenho. Portanto a escolha da IDE deve ser feita com base em sua aplicação e no nível de

experiência do programador, com o objetivo de diminuir as necessidades na hora de solucionar um problema.

5. REFERÊNCIAS BIBLIOGRÁFICAS

NOSCHANG, L. F.; FILLIPI PELZ, E. A.; RAABE, A. Portugol studio: Uma ide para iniciantes em programaçao. Anais do CSBC/WEI, páginas 535–545, 2014.

A. Patrizio. **The history of visual development environments: Imagine there's no ides. it's difficult if you try**. Acessado em 3 out. 2017. Online. Disponível em: https://www.mendix.com/blog/the-history-of-visual-development-environments-imagine-theres-no-ides-its-difficult-if-you-try/

N. Kleina. **A história da Internet: pré-década de 60 até anos 80 [infográfico].** Acessado em 4 out. 2017. Online. Disponível em : https://www.tecmundo.com.br/infografico/9847-a-historia-da-internet-pre-decada-de-60-ate-anos-80-infografico-.htm

Microsoft Corporation. **Microsoft lança o maior conjunto de ferramentas de sua história**. Acessado em 3 out. 2017. Online. Disponível em: https://www.microsoft.com/brasil/pr/devtools.htm