Assinale a área do painel:

( ) Painel 1. Engenharias
(x) Painel 2. Tecnologia e Gestão Estratégica
( ) Painel 3. Governo, Espaço e Política
( ) Painel 4. Educação e Ensino
( ) Painel 5. Ciência, Tecnologia e Sociedade

O IMPACTO DA PROGRAMAÇÃO ORIENTADA A OBJETOS NAS APLICAÇÕES EMPRESARIAIS NAS ÚLTIMAS DÉCADAS NO MERCADO MUNDIAL E O SEU FUTURO

Randú José Costa Aquino Ribeiro[1], Gustavo Oliveira Nascimento[2], Erik Jonatan Martins Viana[3], Murilo Ribeiro Ferreira

RESUMO: O presente artigo tem como intuito analisar o impacto da orientação a objetos nas aplicações empresariais, estudando o seu desenvolvimento histórico, popularização e a sua bibliografia. A metodologia adotada neste estudo foi bibliográfica, na qual foram utilizados autores desde a década de 1990 até a atualidade para estudar o impacto gerado pela POO nessas últimas décadas, com um segmento de análise de mercado. Os resultados obtidos nessa pesquisa apontam uma estabilidade do uso da orientação a objetos nos últimos anos. A OO hoje tornou-se um paradigma indispensável para as aplicações empresariais. Com esse artigo, é proposto que sejam realizados mais estudos sobre a POO, pois com as mudanças tecnológicas podem surgir novos paradigmas.

PALAVRAS-CHAVE: programação orientada a objetos. aplicações empresariais. futuro. tendências de programação.

THE IMPACT OF OBJECT-ORIENTED PROGRAMMING IN BUSINESS APPLICATIONS IN RECENT DECADES ON THE WORLD MARKET AND ITS FUTURE

ABSTRACT: The present article aims to analyze the impact of object orientation in business applications, studying its historical development, popularization and bibliography. The methodology adopted in this study was bibliographic, in which authors from the 1990s to the present day were used to study the impact generated by OOP in recent decades, with an additional market analysis segment. The results obtained in this research point to a stability of the use of object orientation in recent years. OO today has become an indispensable paradigm for business applications. This article proposes that more research on OOP be carried out, as technological changes may lead to new paradigms.

KEYWORDS: object-oriented programming. business applications. future. programming trends.

INTRODUÇÃO

A Programação Orientada a Objetos (POO) surgiu como um paradigma essencial para o desenvolvimento de software empresarial nas últimas décadas, oferecendo uma

[1] Graduando em Bacharelado em Sistemas de Informação – IFSP – São Paulo, aquino.ribeiro@aluno.ifsp.edu.br.
[2] Graduando em Bacharelado em Sistemas de Informação – IFSP – São Paulo, nascimento.gustavo@aluno.ifsp.edu.br.
[3] Graduando em Bacharelado em Sistemas de Informação – IFSP – São Paulo, erik.viana@aluno.ifsp.edu.br.
[4] Graduando em Bacharelado em Sistemas de Informação – IFSP – São Paulo, ribeiro.murilo@aluno.ifsp.edu.br

estrutura mais organizada e modular para sistemas complexos. A POO permite o conceito de encapsulamento de dados e comportamentos em objetos, o que facilita na reutilização de código e torna o software mais flexível e fácil de manter (MEYER, 1997).

Essa abordagem é especialmente eficaz em ambientes de negócios dinâmicos, onde a manutenção contínua e a escalabilidade dos sistemas são cruciais.

Além disso, a POO não apenas melhorou a organização interna dos sistemas, mas também impulsionou a criação de padrões de projeto e frameworks reutilizáveis que padronizam e otimizam o processo de desenvolvimento de software. Estes padrões, como os propostos por Gamma et al. (1994), têm desempenhado um papel vital ao proporcionar consistência e eficiência no desenvolvimento, resultando em um aumento significativo na produtividade das equipes de desenvolvimento e na qualidade final dos produtos lançados no mercado.

Assim, este artigo se propõe a analisar o impacto profundo da POO nas aplicações empresariais, focando nos benefícios em termos de modularidade, reusabilidade e facilidade de manutenção. Além disso, será destacado como a adoção desse paradigma tem permitido que os sistemas empresariais evoluam de forma eficiente, acompanhando as rápidas mudanças tecnológicas e as demandas do mercado global.

FUNDAMENTAÇÃO TEÓRICA

Nos primeiros anos da propagação da Programação Orientada a Objetos no ambiente acadêmico, e por conseguinte, sua adoção nas aplicações empresariais, este paradigma causou divisão de opiniões. Na década de 1990, houve uma cisão entre pesquisadores da área de informática que defendiam veementemente o novo modelo e aqueles que eram céticos em relação a sua eficácia. Esse posicionamento muito se deu por conta das dificuldades de se aprender o novo método e reescrever quase que por inteiro um sistema empresarial que havia sido elaborado com um paradigma diferente (em geral, o paradigma estruturado). (BHATTACHERJEE e GERLACH, 1998)

Ademais, por ser uma novidade (a orientação a objetos), era de pouco conhecimento prático dos programadores daquela época como manuseá-la. Até o ano de 2006, os conceitos da POO ainda não estavam consolidados, e suas definições variavam entre diversos autores e suas abordagens, o que complicava o entendimento daquele, que até o momento, era um novo modelo de programação, tanto para desenvolvedores de início de carreira quanto para veteranos, muitas vezes em posição de senioridade. (ARMSTRONG, 2006)

Apesar dessas dificuldades, linguagens de programação como Java e C++, essencialmente orientadas a objetos, se mantiveram entre as cinco principais linguagens de programação pesquisadas desde que índices de larga escala começaram a ser utilizados, como o índice Tiobe (2024). Em decorrência da evolução do conhecimento acerca destas linguagens e a implementação de sistemas orientados a objetos, diversas aplicações foram desenvolvidas usando seus princípios. A linguagem Java, por exemplo, chamou bastante atenção pelo seu slogan “*Write Once, Run Anywhere*”, ou seja, era uma linguagem suportada em qualquer ambiente que atendesse aos requisitos para a instalação e uso da *Java Virtual Machine* (JVM), como nos servidores web, softwares computacionais, e até em alguns eletrodomésticos. (GOSLING; James, 1997) Isto contribuiu para a popularização desta linguagem e, naturalmente, levou a uma expansão mais rápida da POO e a construção de um ecossistema em torno dela.

Para as empresas, o desenvolvimento da POO até a década de 2010 foi bastante relevante por trazer benefícios práticos às equipes de Tecnologia de Informação, pois facilitou que os grandes sistemas dessas empresas ficassem mais organizados e mais fáceis de se depurar e de fazer a manutenção dos seus códigos. Neste período, o desenvolvimento de aplicações complexas não estava muito ligado aos sites, que, em sua maioria, eram simples e com poucos recursos para os desenvolvedores. Os sistemas complexos e orientados a objetos, em sua maioria, eram usados para o desenvolvimento de

sistemas de ERP, CRM e SGBDs (Sistemas Gerenciadores de Banco de Dados). (FERREIRA, COUTO e MICCHELUCCI, 2011)

Todavia, a partir de 2007, ano no qual foi lançado o primeiro Iphone, iniciou uma revolução tecnológica, modificando completamente o cenário do desenvolvimento de software. Não somente ter um celular se tornou vital para a vida urbana, como eles se provaram serem consideravelmente mais baratos que um computador, o que levou em poucos anos, o termo *mobile first* a ser cunhado pelo diretor de produtos da Google, Luke Wroblewski (2011) em seu livro Mobile First, que se refere a necessidade de se programar priorizando soluções para dispositivos móveis, especialmente no que tange a internet. Nesta nova era, desenvolvedores começaram a se preocupar com questões como responsividade, tempo de uso, aplicativos e maior interação dinâmica entre usuário e sistema, pois essa nova tecnologia exigia novas formas de interação com o usuário, por ser um dispositivo multitarefas de fácil transporte. Junto a isso, com o desenvolvimento de microprocessadores mais rápidos, as aplicações web e mobile tiveram espaço para crescer em tamanho e complexidade, tornando indispensável a adoção de uma série de boas práticas e a solidificação de linguagens robustas, capazes de lidar com abstrações do mundo real.

Em virtude dessas mudanças, foram usadas principalmente as linguagens de programação Java para desenvolvimento Android (com a criação do Kotlin posteriormente), e C# para o desenvolvimento IOS, e ambas são linguagens essencialmente orientadas a objetos. Essa adoção aconteceu por conta da versatilidade da linguagem Java, como mencionado anteriormente, e pela compatibilidade entre os diferentes sistemas operacionais que o C# conseguia abranger. No tocante da melhoria dos microprocessadores que permitiram o crescimento da complexidade das aplicações, a POO embarcada nesses sistemas também ajudaram a organizar o código, mediante os seus princípios de encapsulamento e herança, para que fosse possível uma manutenção mais facilitada do código dessas aplicações. Dessa maneira, as equipes de desenvolvimento das empresas puderam montar algoritmos mais complexos e que conseguiam ser atualizados com maior facilidade e rapidez.

Apesar de atualmente o C# e Java não serem mais as principais linguagens de programação (STACKOVERFLOW; 2024), a naturalização e consolidação da POO no ambiente empresarial também trouxe um entendimento amplo sobre como programadores deveriam encarar a construção de software. Por exemplo, adotou-se o princípio de que modelagem é uma parte fundamental deste processo (ANICHE; KON e YODER, 2019), e unificar conceitos entre diferentes implementações é necessário, o que levou a criação da UML (RUMBAUGH, JACOBSON e BOOCH, 1998, p.5-6) e, posteriormente, a adoção de arquiteturas como a MVC para auxiliar a construção de aplicações bem estruturadas. (SUNDAY e ELUGWU, 2022)

No entanto, nos dias atuais, os requisitos aos quais uma aplicação empresarial precisa atender mudaram significativamente. Não somente programadores começam um projeto já se utilizando de uma pletora de *frameworks* e *libraries* com implementações amplamente diferentes e que podem modificar drasticamente a forma com que a linguagem de programação e seus paradigmas são usados, como as aplicações finais precisam ser heterogêneas e atuar, muitas vezes, no nível de microsserviços, que serão consumidos por serviços desconhecidos. (ANICHE; KON e YODER, 2019)

O cenário atual é bem diferente das duas décadas anteriores, e apesar do paradigma de orientação a objetos continuar sendo a escolha principal para programadores em diferentes ramos (TRIAJI, PRATOMO e DP, 2021), este novo cenário leva a mudanças na forma que POO é implementado e pensado.

METODOLOGIA DA PESQUISA

O presente estudo sobre os modelos de Programação Orientada a Objetos está focado em seus impactos nas aplicações empresariais nas últimas décadas no mercado mundial. Esta pesquisa foi dividida em duas partes, sendo a primeira um estudo bibliográfico e a segunda uma análise de mercado.

Na primeira parte, prepondera o estudo bibliográfico, construído a partir da análise de artigos e livros da década de 90 até o começo da década atual, obtendo uma visão clara da evolução do paradigma e dos problemas atuais que ele enfrenta, ao mesmo tempo que constroi um contraste que permite a resposta ou pelo menos a reflexão em cima de perguntas como por exemplo, quais ideias mudaram, e o que sobreviveu ao teste do tempo. As fontes usadas neste segmento foram obtidas pelo Google Académico, de diferentes universidades e autores, e publicado em prestigiados veículos científicos da área, como IEEE e Elsevier. A metodologia utilizada foi identificar textos alinhados diretamente com o tema da pesquisa, usando palavras-chave como *OOP adoption in companies*, *OOP evolution in companies*, *OOP adoption in business* e *Difficulties on programming language adoption*. Em seguida, o resumo e introdução de cada artigo foi lido, refinando a lista de referências pela primeira vez. Em uma segunda leitura, os artigos restantes foram fixados a partir de uma leitura crítica e apurada, permitindo com que mais artigos fossem descartados. No fim, os artigos escolhidos foram organizados e a redação da pesquisa foi realizada.

Na segunda parte, duas pesquisas anuais da área de programação foram agrupadas e analisadas com o intuito de entender as tendências atuais do mercado, e comparar com o estudo bibliográfico realizado previamente. Elas foram selecionadas a partir de uma série de critérios, incluindo volume de dados (quantidade de pessoas que participaram da pesquisa / quantidade de dados coletados), relevância da organização pesquisadora (ela é conhecida no mercado de programação? ela tem reputação positiva?), qualidade das perguntas (o quão relevante elas são para a análise que este artigo busca) e viés (o quão enviesada estas pesquisas são e o quão aberta elas são em relação a isso).

RESULTADOS E DISCUSSÃO

Duas pesquisas foram analisadas: *The State of Developer Ecosystem* da Jetbrains (2023), conhecida pelo seu IDE para programação Java e Kotlin, IntelliJ IDEA e *Stack Overflow Annual Developer Survey*, do Stack Overflow (2024), o maior fórum de programação do mundo. Considerando que estas pesquisas possuem grande fluxo de informações, duas perguntas focais, adequadas aos propósitos deste trabalho foram escolhidas:

Pergunta 1: Como linguagens de programação tipicamente OO se encontram em termos de popularidade e uso? (Java, C#)

Pergunta 2: Programadores estão adotando novas tecnologias? Se sim, quais? Como isto se reflete na popularidade de certas linguagens? E como isto reflete uma possível mudança do mercado?

Dados Extraídos	Pesquisa (Jetbrains - 2017)	Pesquisa (Stack Overflow - 2017)	Pesquisa (Jetbrains - 2023)	Pesquisa (Stack Overflow - 2023)
Popularidade do Java	30%	40%	49%	30,5%
Popularidade do C#	20%	34,1%	21%	27,6%
Popularidade do Python	32%	32%	54%	49,3%
Popularidade do Rust	-	-	10%	13%
Popularidade do JS	65%	62,5%	61%	63,6%
Adoção de IA no Fluxo de Trabalho	-	-	77% (ChatGPT)	43%
Trabalho com Big Data/Machine Learning.	16%	8,4%	9%	3,7%

Trabalho com Microsserviços	-	-	34%	Ausente
Trabalho com DevOps	-	11,1%	11%	1,8%

Fonte: os autores.

Partindo destas perguntas e dos dados coletados, observa-se alguns fenômenos: Java, ao longo dos anos, perdeu popularidade e C# estagnou. Em 2017, Java era a terceira linguagem de programação mais usada na pesquisa da Jetbrains e do Stack Overflow, enquanto em 2023, Java figurava em quinto e sétimo respectivamente, enquanto C# na pesquisa da Jetbrains caiu da quarta posição em 2017 para a nona posição em 2023, que é a posição que a linguagem se encontra na pesquisa do Stack Overflow. Por outro lado, Python saiu da quinta posição nas duas e foi para segunda e terceira posição respectivamente e Javascript manteve sua hegemonia, não caindo de posição de 2017 a 2023. Além destas linguagens, Rust, apesar de sua adoção pequena (10% na pesquisa do Jetbrains e 13% na pesquisa do Stack Overflow), é considerada a linguagem mais provável de programadores continuarem usando na pesquisa do Stack Overflow e é a linguagem mais provável de programadores adotarem na pesquisa do Jetbrains.

Em termos de tecnologias, Inteligência Artificial foi uma categoria emergente em ambas pesquisas no ano de 2023, e seus dados revelam que não só mais programadores pretendem adotar IA em seu fluxo de trabalho ou trabalhar diretamente com desenvolvimento de IA como uma parcela significativa (43% na pesquisa do Stack Overflow e 77% na pesquisa do Jetbrains) já o fazem. Em adição a isso, Big Data e Machine Learning são a realidade de uma parcela considerável de programadores nas duas pesquisas (mesmo após sofrerem uma pequena queda), e na pesquisa do Jetbrains, 34% dos programadores relataram trabalhar com microsserviços.

Os resultados, quando cruzados, traçam um cenário para o mercado onde linguagens como Java e C# estão perdendo popularidade, mas continuam relevantes, enquanto linguagens com paradigmas e conceitos diferentes estão ganhando tração. Python teve o crescimento mais expressivo entre todas as linguagens analisadas, e sendo uma linguagem conhecida por ter um ecossistema focado em análise de dados e machine learning, indica como estes assuntos são uma tendência dos últimos anos. Além disso, os números expressivos em relação à inteligência artificial mostram que a área da programação está sofrendo mudanças significativas. Estes resultados corroboram com o que foi apresentado ao longo do trabalho: a Programação Orientada a Objetos nunca perdeu relevância, sempre se adaptando ao longo das diferentes eras e isso se conecta a capacidade deste paradigma de representar situações, trazendo estabilidade ao programador, independente das ferramentas que ele precisa para resolver aquele problema.

No entanto, fica claro também que existem várias tendências de mercado em rápida expansão. Big Data, Machine Learning, IA levam programadores a adotarem outras tecnologias, o que explica a incerteza se este paradigma, e por consequência, as linguagens que o adotam vão continuar sendo relevantes. Existe uma necessidade identificada do paradigma e das linguagens que o utilizam evoluírem e se adaptarem aos tempos modernos.

CONCLUSÕES

Durante a realização do trabalho, a questão inicial que orientou a pesquisa foi: “Orientação a Objetos continua relevante? Se sim, como convencer empresas a adotá-la?” Porém, ao longo da pesquisa, ficou claro que a pergunta errada havia sido feita. Ao invés de nos perguntarmos se ela continua relevante, a pergunta real é: como ela pode continuar sendo relevante e se renovar a esta nova época?

Não existem respostas claras a isso, uma vez que estamos presenciando os primeiros efeitos destas mudanças. As pesquisas continuam indicando interesse em OO mas o seu futuro pode estar desafiado.

E exatamente por isso, propomos que mais pesquisas sejam feitas, tanto para entender mais profundamente os impactos das novas tecnologias na POO como para desenvolver novas formas de usar e enxergar este paradigma, de modo que este modelo possa sobreviver a esta nova era de mudanças e revoluções constantes na programação.

REFERÊNCIAS

FERREIRA, Adhemir; COUTO, Celso; MICCHELUCCI, Andrea. **Aquisição de Sistemas ERP: uma análise dos resultados obtidos pelas empresas**. *Gestão Contemporânea*, n. 9, 2011.

ANICHE, Maurício; YONDER, Joseph W.; KON, Fabio. **Current Challenges in Practical Object-Oriented Software Design.** In: INTERNATIONAL CONFERENCE ON SOTFWARE ENGINEERING: NEW IDEAS AND EMERGING RESULTS, 41., 2019, Montreal. Proceedings [...] IEEE, 2019. p. 113-116.

GAMMA, E.; HELM, R.; JOHNSON, R.; VLISSIDES, J. **Design Patterns: Elements of Reusable Object-Oriented Software**. 1. ed. Boston: Addison-Wesley, 1994. 395 p.

STACK OVERFLOW. **Developer Survey 2024**. Disponível em: https://survey.stackoverflow.co/2024/technology/. Acesso em: 27 ago. 2024.

GitHut 2.0. Disponível em: https://madnight.github.io/githut/#/pull_requests/2024/1. Acesso em: 27 ago. 2024.

C., Agu S.; F., Elugwu. **Object Oriented Programming Approach: A Panacea for Effective Software Development**. African Journal of Advanced Sciences & Technology Research, v. 6, n. 1, p. 1-14, set. 2022.

MEYER, B. **Object-Oriented Software Construction**. 2. ed. New Jersey: Prentice Hall, 1997. 1250 p

TRIAJI, Bagas; PRATOMO, Cucut Hariz; DP, Bambang Purnomosidi. **Programmer's Perspective of Object Oriented Programming (OOP) in Software Development Using Correlation Analysis.** Sintech Journal, Indonesia, 2021, v. 4, n. 1, p. 79-87.

Stack Overflow Annual Developer Survey. Stack Overflow, 2024. Disponível em: https://survey.stackoverflow.co/. Acesso em: 27 ago. 2024

GOSLING, James. **The Feel of Java.** *IEEE*, Califórnia, 1997. p. 53-57.

TIOBE. **TIOBE Index**. Disponível em: https://www.tiobe.com/tiobe-index/. Acesso em: 27 ago. 2024.

ARMSTRONG, Deborah J. **The Quarks of Object-Oriented Development**. *Communications of the ACM*, 2006.

The State of Developer Ecosystem 2023. Jetbrains, 2023. Disponível em: https://www.jetbrains.com/lp/devecosystem-2023/. Acesso em: 27 ago. 2024.

RUMBAUGH, James; JACOBSON, Ivar; BOOCH, Grady. ***The Unified Modeling Language Reference Manual***. *Addison-Wesley*, 1999. 5-6 p.

BHATTACHERJEE, Anol; GERLACH, James. **Understanding and managing OOT adoption**. *Journal of Management Information Systems*, v. 15, n. 2, p. 123-144, 1998.