QUAL MOTIVO DE ALGUNS PROGRAMADORES NÃO UTILIZAREM UML, COMO ELA OS AUXILIA

André Luis da Silva[1], Fabrício Gustavo Henrique[1]

[1]Faculdade de Tecnologia de FATEC Ribeirão Preto (FATEC)

Ribeirão Preto, SP – Brasil

andre.silva316@fatec.sp.gov.br,fabricio.henrique@fatec.sp.gov.br

Resumo. *Este artigo descreve um estudo sobre a utilização da linguagem de modelagem, Linguagem Unificada de Modelagem (UML). Este trabalho apresenta uma análise de duas perguntas quais são os motivos para alguns programadores não utilizarem a UML na modelagem de Software como a linguagem de modelagem UML é capaz de auxiliar os programadores. A pesquisa é baseada na pesquisa bibliográfica e documental. Verificou-se neste trabalho que alguns programadores preferem não usar UML e começar pela sua codificação sendo os diagramas confusos e complexos. A UML pode auxiliar esses programadores a se comunicar e entender o programa corretamente sem ambiguidade no modelo, com os diagramas padronizados.*

Abstract. *This article describes a study on the use of the modeling language, Unified Modeling Language (UML). This work presents an analysis of two issues that are the reasons for some programmers not to use UML in Software modeling, since the UML modeling language is able to help programmers. The research is based on bibliographic and documentary research. It was found in this work that some programmers prefer not to use UML and start with its coding, the diagrams being confusing and complex. UML can help these programmers to communicate and understand the program correctly without ambiguity in the model, with standardized diagrams.*

1. Introdução

Os programadores programam novos sistemas e softwares utilizando a orientação a objetos um problema é o fato de não existir uma notação padronizada e eficaz que possa abranger qualquer tipo de aplicação transformando com isso a escolha do método uma decisão extremamente importante que leva a discussões sobre o método mais avançado e adequado para ser utilizado. Com isso pode-se entender a importância da UML, usada para unificar os modelos de maneira padronizada e eficiente, sendo UML de propósito geral e pode ser trabalhada a todos os domínios e aplicação de sistemas.

A UML (Linguagem Unificada de Modelagem) é uma linguagem gráfica de modelagem orientada a objetos para visualização, especificação, construção e documentação para desenvolver sistemas computacionais de propósito gerais que pode ser aplicada a todos os softwares e que permite demonstra-lo de forma padronizada (BOOCH; RUMBAUGH; JACOBSON, 2012). A UML baseia-se em diagramas de modelagem visual, e se propõe ser uma boa modelagem de sistemas orientada a objeto, pois é uma modelagem unificada e os envolvidos no desenvolvimento do sistema tem uma facilidade de interpretação no projeto, sendo adotada por muitas empresas.

Todo e qualquer programa deve ser modelado antes de se iniciar sua codificação, para evitar o retrabalho e possíveis desvios de metas de projetos de sistemas. A UML é um jeito universal para programadores e que os auxilia a se comunicar e programar o software em diagramas padrões, facilitando o entendimento do escopo dos sistemas.

Os programadores de sistemas constroem sistemas consistentes e que atendem às necessidades de seus usuários. Sem dúvida, ter uma equipe de programadores articulados é parte fundamental da solução do problema e esse é um aspecto que, algumas vezes é negligenciado na modelagem de sistemas. Muitas vezes as empresas e programadores não dão ênfase a essa fase do modelo, e cometem alguns erros de análise e modelagem, pois os projetos começam nas fases de codificação (RIBEIRO, 2012).

Contudo alguns programadores afirmam que conseguem atingir todas as necessidades de um sistema de informação sem modelagem UML. Eles falam que não precisam de diagramas UML para os projetos, sendo os diagramas confusos.

A pesquisa tem o intuito de responder ao problema de pesquisa: Porque alguns programadores não utilizam UML na modelagem de Software?

O trabalho aqui apresentado tem como objetivo descrever como a utilização da UML contribui para os projetos de software e porque alguns programadores não utilizam a UML.

A metodologia utilizada foi a pesquisa bibliográfica e a pesquisa documental com uma abordagem qualitativa.

2. Referencial Teórico

Nesta seção são discutidos temas relacionados à UML e modelagem de sistemas de forma geral.

2.1 Introdução a UML

A UML, (Unfied Modeling Language) é uma linguagem-padrão para a elaboração de estrutura de projetos de software. Ela poderá ser empregada para a visualização, especificação, construção e documentação de artefatos de sistemas (BOOCH; RUMBAUGH; JACOBSON, 2012). A modelagem UML é apresentada em diagramas padronizados sendo utilizada para uma melhor compreensão do software programado, a UML define um numero de diagramas que permite focar para aspectos diferentes de cada software ou projeto, com o emprego certo, os diagramas auxiliam no seu entendimento e interpretação do software que esta sendo codificado.

De acordo com (GUEDES, 2011) a UML é uma linguagem de modelagem de proposito geral que pode ser aplicada a todos os domínios e aplicação. Essa linguagem tornou-se, nos últimos anos, a linguagem-padrão de modelagem adotada internacionalmente pela indústria de engenharia de software.

A UML é uma unificação bastante boa que facilita o processo de desenvolvimento de softwares, pois permite uma maior comunicação e entendimento para obter um software de maior qualidade e com uma boa produtividade.

A UML é necessária, pois os sistemas mudam constantemente. E a UML também é necessária para que desenvolvedores visualizem os produtos de seus trabalhos em diagramas padronizados, utilizando símbolos e padrões, tornando uma melhor

comunicação e não ambígua entre os participantes do projeto com relação aos detalhes do comportamento do sistema. Junto com uma notação gráfica, a UML também especifica significados, isto é, a semântica (GUEDES, 2011).

2.2 Por que modelar

Um modelo é uma simplificação da realidade. Os modelos fornecem a planta do projeto de um sistema. Os modelos poderão abranger planos detalhados, assim como planos mais gerais com uma visão panorâmica do sistema considerado. Um bom modelo inclui aqueles componentes que têm ampla repercussão e omite os componentes menores que não são relevantes em determinado nível de abstração. Todos os sistemas podem ser descritos sob diferentes aspectos, utilizando modelos distintos e cada modelo será, portanto, uma abstração semanticamente específica do sistema (BOOCH; RUMBAUGH; JACOBSON, 2012).

Por mais simples que seja, todo sistema deve ser modelado antes de se iniciar sua implementação, porque os sistemas de informação costumam ter tendência de complexidade e abrangência. Alguns programadores falam que sistemas de informação são dinâmicos, pois os clientes e os mercados desejam modificações ou melhorias no sistema (GUEDES, 2011).

2.3 Uso da UML e de seus diagramas

O jeito de descrever os vários aspectos da modelagem é pelo uso de diagramas. O diagrama é uma apresentação gráfica de um colecionado de elementos representados por um gráfico conectado de item e relacionamento. A UML define um número de diagramas que permite focar para aspectos diferenciados de cada sistema de maneira à demostrar o sistema programado (BOOCH; RUMBAUGH; JACOBSON, 2012).

Os diagramas usados para exemplificar o uso na UML neste trabalho: diagrama de caso de uso, diagrama de classe, diagrama de sequencia e diagrama de atividade.

Os diagramas de caso de uso são um dos diagramas usados na linguagem de modelagem UML, com o foco na visão, documentação, comportamento do sistema e na especificação do sistema. Esses diagramas fazem com que sistemas e subsistemas, sejam melhores de entender, por sua visão mais externa dos elementos do software (BOOCH; RUMBAUGH; JACOBSON, 2012).

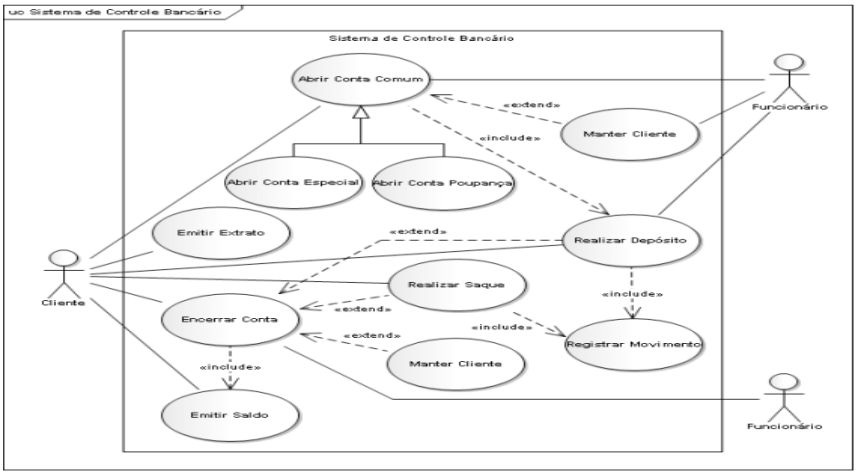

Figura 1. Exemplo de diagrama de caso de uso

Fonte: (GUEDES, 2011)

O diagrama de classe é provavelmente o mais utilizado e importante da UML. Serve de base para a maioria dos diagramas. Como o próprio nome diz, define a estrutura das classes utilizadas pelo sistema, com os atributos e métodos e o relacionamento das classes existentes no software (GUEDES, 2011). Exemplo sistema de controle bancário.

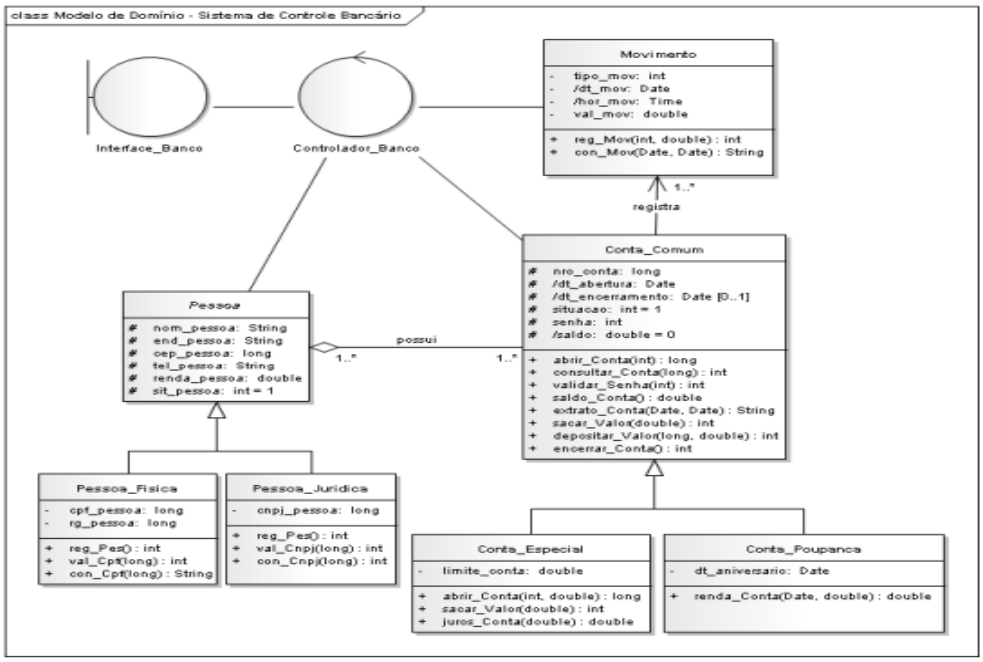

Figura 2. Exemplo de diagrama de classes

Fonte: (GUEDES, 2011)

Os diagramas de sequencia, chamados de diagramas de interação mostra uma interação, compostas de um conjunto de relacionamento e seu objeto, incluindo as mensagens que são trocadas entre esses objetos. O objetivo do diagrama de sequencia e descrever, no tempo, a composição de mensagens (BOOCH; RUMBAUGH; JACOBSON, 2012).

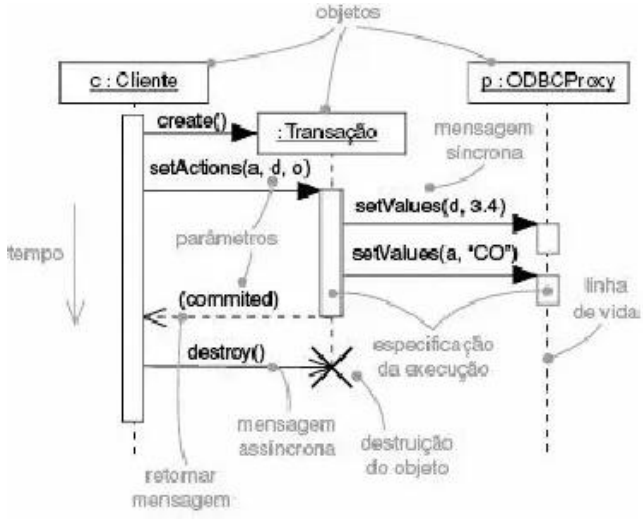

Figura 3. Exemplo de diagrama de sequencia

Fonte: (BOOCH; RUMBAUGH; JACOBSON, 2012)

O diagrama de atividades é composto de um gráfico que representa fluxo de controle de uma atividade e um foco em coordenar o comportamento de sistema. Sendo diferente do fluxo tradicional, o diagrama de atividade exibe a sua concorrência de controle (BOOCH; RUMBAUGH; JACOBSON, 2012).

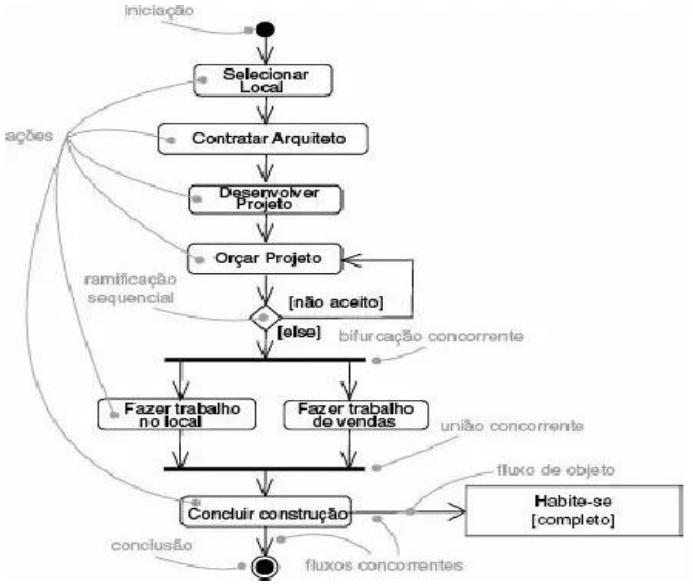

Figura 4. Exemplo de diagrama de atividade

Fonte: (BOOCH; RUMBAUGH; JACOBSON, 2012)

A UML é usada no desenvolvimento dos mais diversos tipos de sistemas. Ela abrange quaisquer características de um sistema em um de seus diagramas e pode ser aplicada em diferentes fases do desenvolvimento de um sistema, desde a especificação da análise de requisitos até a finalização, com a fase de testes (GUEDES, 2011).

A UML é adequada para a modelagem de sistemas, cuja abrangência poderá incluir desde sistemas de informação corporativos a serem distribuídos; aplicações baseadas na Web e até sistemas complexos embutidos de tempo real. É uma linguagem muito expressiva, abrangendo todas as visões necessárias ao desenvolvimento e implantação desses sistemas (BOOCH; RUMBAUGH; JACOBSON, 2012).

A UML emprega notações e regras que torna possível expressar modelos. Mas ela não prescreve como o trabalho deve ser feito, ou seja, não tem um processo de como o trabalho deve ser desenvolvido, já que a UML foi desenvolvida para ser usada em diversos métodos de desenvolvimento (GUEDES, 2011).

Para o uso da UML com sucesso é necessário adotar algum tipo de método de desenvolvimento, especialmente em sistema de grande porte, no qual a organização de tarefa é essencial. A utilização de um processo de desenvolvimento torna mais eficiente calcular o progresso do projeto, controlar e melhorar o trabalho (GUEDES, 2011).

A UML resolveu o problema de falta de padronização da notação de modelagem de um sistema orientado a objetos, já que antes da padronização, foram propostos vários métodos orientados a objetos no período de fragmentação, com notações próprias e isto dificultava a comunicação entre analistas ou projetistas de organizações diferentes que usavam métodos diferentes com notações de modelagem de sistemas orientados a objetos diferentes (GUEDES, 2011).

2.4 Desuso da UML

Uma das principais discussões do trabalho se concentra no entendimento da correspondência dos motivos que alguns programadores não usam a UML.

Com isso realizou-se uma pesquisa dos motivos dos entraves do uso da UML, e quais eram as dificuldades pelos profissionais de tecnologia na UML.

Uma das desvantagens da UML é que gasta mais tempo modelando software em relação gerencia dos diagramas. Para que a UML ofereça um bom trabalho, um programador precisa sincronizar os códigos que requerem a manutenção e o tempo de um projeto de software e que também adiciona trabalho a um projeto de desenvolvimento de software (PCZONE, 2020). Pequenas empresas e desenvolvedores independentes podem não ser bons com a quantidade adicional de trabalho necessária para sincronizar o código.

Não é necessário utilizar UML para comunicar os projetos. Os diagramas informais fazem o mesmo que a UML (OLIVER, 2020).

Nem sempre está claro quem se beneficia com um diagrama UML. De acordo com um artigo publicado no site da Eiffel Software, a UML não é vantajosa para os desenvolvedores de software, principalmente porque os desenvolvedores de software trabalham com código, não com diagramas ou imagens. Os diagramas UML para gerentes de projeto ou executivos para ilustrar como uma ferramenta de software funcionará, mas pode ser mais fácil desenhar o diagrama em um quadro branco ou folha de papel, em vez de dedicar algum tempo para aprender a linguagem UML (TECHWALLA, 2020).

3. Materiais Métodos

Considerando os procedimentos metodológicos, realizou-se uma pesquisa bibliográfica e documental, sendo que a problemática foi abordada de forma qualitativa.

Na pesquisa bibliográfica (leitura realizada em agosto a outubro), primeiramente foi estudada a linguagem UML, utilizando os livros de Booch, Rumbaugh e Jacobson (2012) e Guedes (2011), e foram estudadas as funcionalidades e o sequenciamento dos artefatos presentes na linguagem UML. Também se utilizou esses livros para entender o porquê empregar a UML é relevante para o projeto e desenvolvimento de software, bem como o porquê fazer modelagem de software.

Para desenvolver um bom software, é necessária uma modelagem para construir modelos para a visualização, controlar a arquitetura do software e compreender o software, auxiliando na simplificação e reaproveitamento. Com isto, o desenvolvimento de um software bom se torna uma questão de arquitetura, processos e ferramentas (BOOCH; RUMBAUGH; JACOBSON, 2012).

Qualquer projeto é beneficiado pelo uso de modelagem, sendo ela útil na visualização do software e planejamento auxiliando na construção do software correto (BOOCH; RUMBAUGH; JACOBSON, 2012).

O modelo é utilizado de acordo com a sua visualização em relação a sistemas de software. Para um programa orientado a objeto, o modelo trabalhado será com várias classes de sistema, cada modelo sendo expresso dependendo do seu sistema de software e natureza (BOOCH; RUMBAUGH; JACOBSON, 2012).

A UML é empregada para a visualização, a especificação, documentação, e construção de sistemas de software, sendo adequada para modelagem de sistemas, que abrange sistema de informação e sistemas baseada em web, sendo uma linguagem expressiva que abrange diversas visões desses softwares, não sendo difícil compreender e usar UML (BOOCH; RUMBAUGH; JACOBSON, 2012).

Os desenvolvedores do sistema podem empregar a UML para escrever modelos, e qualquer outro desenvolvedor, ou até outras ferramentas, pode interpretá-los (BOOCH; RUMBAUGH; JACOBSON, 2012).

A modelagem de um software é fazer modelos de software. Estes modelos de software são a visão do sistema, uma abstração com propósito de descrever aspectos estruturais e comportamentais do software. Dessa maneira, o modelo de casos de uso, por exemplo, fornece uma visão dos requisitos do sistema (GUEDES, 2011).

A modelagem de software é boa para diminuir os custos com a manutenção do software e auxilia a compreender os sistemas corretamente e a outros programadores interpretar esse sistema e mantê-lo (GUEDES, 2011).

A UML, linguagem visual utilizada para modelagem de softwares, é empregada pelas indústrias de engenharia de software e amplamente usada (GUEDES, 2011).

A UML emprega notações, semântica e regras que expressa modelos de software. Mas ela não prescreve como o trabalho deve ser feito, ou seja, não tem um processo de como o trabalho deve ser desenvolvido, já que a UML foi desenvolvida para ser usada em diversos métodos de desenvolvimento (GUEDES, 2011).

Na pesquisa documental (leitura realizada em agosto a outubro), utilizando os artigos de Oliver (2020) e Simon (2017). Também se utilizou esses artigos para entender o porquê alguns programadores não utilizam a UML.

Não precisa de UML para comunicar seus projetos. Pode-se ter o mesmo impacto e efeito com diagramas informais de caixa e linha criados no PowerPoint, ou em um quadro. Como a codificação é uma linguagem formal por si só, muitos desenvolvedores não preferem a complexidade e a formalidade no nível arquitetônico, o que desencoraja o uso de UML (OLIVER, 2020).

Alguns programadores "apenas usam caixas e linhas em um quadro" como uma forma de comunicar ideias (SIMON, 2017).

Nem sempre está claro quem se beneficia com um diagrama UML. De acordo com um artigo publicado no site da Eiffel Software, a UML não é vantajosa para os desenvolvedores de software, principalmente porque os desenvolvedores de software trabalham com código, não com diagramas ou imagens. Os diagramas UML para gerentes de projeto ou executivos para ilustrar como uma ferramenta de software funcionará, mas pode ser mais fácil desenhar o diagrama em um quadro branco ou folha de papel, em vez de dedicar algum tempo para aprender a linguagem UML (TECHWALLA, 2020).

Alguns programadores acham que é melhor não utilizar UML porque ele pode fica confuso, fazendo com que os novos programadores fiquem fora do aprendizado ou de usá-lo. Isso torna o uso da UML menos importante por causa de seu tamanho de diagramas e códigos (PCZONE, 2020).

Não está claro quem se beneficia com um diagrama UML, sendo os diagramas

opressores e difíceis (JANETPANIC, 2020).

4. Conclusão

O trabalho proposto pretendeu responder qual motivo de alguns programadores não utilizarem UML, como ela os auxilia, sendo relevante para entender os motivos pelos quais alguns programadores não utilizarem a UML e contemplou como o uso da UML pode auxiliar esses programadores que não empregam a UML em seus softwares. Como que a UML é uma unificação boa, universal e que facilitará os programadores no seu processo de desenvolvimento de seus softwares, pois permite uma maior comunicação e entendimento nas pessoas envolvidas neste processo e essa modelagem pode agregar as necessidades desses programadores, que não a utilizam, auxiliando-os em seus projetos de softwares.

A problemática abordada no trabalho é porque alguns programadores não utilizam UML na modelagem de Software? Em virtude dessa problemática, analisaram que os programadores conseguem todas suas necessidades sem utilizar a UML, os diagramas informais fazem o mesmo que a UML sem a necessidade de aprender vários diagramas complexos.

O trabalho apresentado tem como objetivo explicar como a utilização da UML contribui para os projetos de software e interpretar porque alguns programadores não utilizam a UML.

Respondendo o primeiro, a UML trará uma comunicação muito grande aos desenvolvedores, pois a organização da modelagem em visão e a divisão de diagramas especificando características estáticas e dinâmicas do sistema torna-se mais fácil de ser utilizada e fará com que qualquer tipo de comportamento seja visualizado em diagramas. Facilitará o desenvolvimento de software, devido a maior comunicação e aproveitamento dos modelos desenvolvidos pelos seus vários analistas. Tendo assim software de maior qualidade e uma excelente produtividade devido aos recursos oferecidos pela UML.

O segundo respondido, os programadores preferem não utilizar a UML começando os projetos de software pela codificação, sendo os diagramas confusos ou complexos, e conseguem adquirir todas as necessidades de um sistema de informação sem modelagem, é mais fácil desenhar o diagrama em um quadro branco ou folha de papel, em vez de dedicar algum tempo para aprender a linguagem UML. Não é vantajosa para os desenvolvedores de software, principalmente porque os desenvolvedores de software trabalham com código, não com diagramas.

Conclui-se que para um bom software, o programador precisa utilizar alguma ferramenta como a UML que faz com o software tenha uma boa produtividade, robustez no entendimento da compreensão do modelo programado e os diagramas padronizados auxiliam no seu entendimento e interpretação do software que está sendo programado e para evitar o retrabalho e possíveis desvios de metas no sistema programado e a UML é necessária, pois os sistemas mudam constantemente precisando de uma modelagem adequada e unificada sendo de fácil visualização e comunicação em relação à equipe de programadores.

Referências

BOOCH, G; RUMBAUGH, J e JACOBSON, I: UML, Guia do Usuário: tradução; Fábio Freitas da Silva, Rio de Janeiro, Campus, 2012.

GUEDES, G. T. UML 2: uma abordagem prática. Novatec, 2011.

JANETPANIC, Janetpanic: What are disadvantages of UML?, 2020. Página inicial. Disponível em: < https://janetpanic.com/what-are-disadvantages-of-uml/#What_are_the_advantages_and_disadvantages_of_using_UML/>. Acesso em: 18 ago. de 2021.

OLIVER, Rachel. Creately: Why the Software Industry Has a Love Hate Relationship with UML Diagrams, 2020. Página inicial. Disponível em: <https://creately.com/blog/diagrams/advantages-and-disadvantages-of-uml/>. Acesso em: 4 de nov. de 2020.

PCZONE, Pczone: 4 Advantages and Disadvantages of UML Diagrams for Companies, 2020. Página inicial. Disponível em: <https://www.pczone.co.uk/4-advantages-and-disadvantages-of-uml-diagrams-for-companies/>. Acesso em: 20 ago. de 2021.

RIBEIRO, Leandro. O que é UML e Diagramas de Caso de uso: Introdução Prática à UML. DEVMEDIA, 2012. Disponível em: <https://www.devmedia.com.br/o-que-e-uml-e-diagramas-de-caso-de-uso-introducao-pratica-a-uml/23408>. Acesso em: 24 de out. de 2020.

SIMON, Brown. DEV: You don't need to use UML, 2020. Página inicial. Disponível em: <https://dev.to/simonbrown/you-don-t-need-to-use-uml-4kaa>. Acesso em: 4 de nov. de 2020.

TECHWALLA. Techwalla: The Disadvantages of UML, 2021. Página inicial. Disponível em: <https://www.techwalla.com/articles/the-disadvantages-of-uml >. Acesso em: 20 de out. de 2020.