CENTRO UNIVERSITÁRIO DAS FACULDADES METROPOLITANAS UNIDAS

BRUNO ASCENÇÃO PEREIRA – RA: 1889314

PROJETO DE ESTÁGIO – UML

São Paulo – SP

2021

SUMÁRIO

1. UNIFIED MODELING LANGUAGE

Unified Modeling Language (UML) é uma linguagem de modelagem padronizada que consiste em um conjunto integrado de diagramas, desenvolvido para ajudar os desenvolvedores de sistema e software a especificar, visualizar, construir e documentar os principais artefatos de sistemas de software, e também para modelagem de negócios e outros sistemas que não sejam de software. A UML representa uma coleção das melhores práticas de engenharia que provaram ser bem-sucedidas na modelagem de sistemas grandes e complexos. É uma parte muito importante do desenvolvimento de software orientado a objetos e do processo de desenvolvimento de software. A UML usa principalmente notações gráficas para expressar o design de projetos de software. O uso da UML ajuda as equipes de projeto a se comunicarem, ter uma visão geral de como vai funcionar um sistema, explorar projetos em potencial e validar a arquitetura de um software.

2. A ORIGEM DO UML

(Visual Paradigm 2021) UML é uma notação que resultou da unificação de OMT de:

1. Object Modeling Technique OMT [James Rumbaugh 1991] - era a melhor para análise e sistemas de informação com uso intensivo de dados.

2. Booch [Grady Booch 1994] - foi excelente para design e implementação. Grady Booch havia trabalhado extensivamente com a linguagem Ada e foi uma pessoa importante no desenvolvimento de técnicas orientadas a objetos para a linguagem. Embora o método Booch fosse forte, a notação foi menos bem recebida.
3. OOSE (Object-Oriented Software Engineering [Ivar Jacobson 1992]) - apresentava um modelo conhecido como Casos de Uso. Os casos de uso são uma técnica poderosa para entender o comportamento de um sistema inteiro.

Em 1994, Jim Rumbaugh, o criador da OMT, surpreendeu o mundo do software quando deixou a General Electric e se juntou a Grady Booch na Rational Corp. O

objetivo da parceria era fundir suas ideias em um método único e unificado (o título provisório do método era de fato o "Método Unificado").

Em 1995, o criador do OOSE, Ivar Jacobson, também se juntou à Rational, e suas ideias (particularmente o conceito de "Casos de Uso") foram alimentadas no novo Método Unificado - agora chamado de Linguagem de Modelagem Unificada. A equipe de Rumbaugh, Booch e Jacobson é carinhosamente conhecida como os "Três Amigos"

3. HISTÓRIA DA UML

(Visual Paradigm 2021) Durante 1996, a primeira solicitação de proposta (RFP) emitida pelo Object Management Group (OMG) forneceu o catalisador para que as organizações unissem forças para produzir uma resposta conjunta à RFP.

A Rational estabeleceu o consórcio UML Partners com várias organizações dispostas a dedicar recursos para trabalhar em direção a uma definição UML 1.0 forte.

Essa colaboração produziu a UML 1.0, uma linguagem de modelagem bem definida, expressiva, poderosa e de aplicação geral. Isso foi enviado ao OMG em janeiro de 1997 como uma resposta inicial à RFP.1

(Visual Paradigm 2021) Em janeiro de 1997, a IBM, ObjecTime, Platinum Technology, Ptech, Taskon, Reich Technologies e Softeam também enviaram respostas de RFP separadas ao OMG. Essas empresas se juntaram aos parceiros UML para contribuir com suas idéias e, juntos, os parceiros produziram a resposta UML 1.1 revisada. O foco do lançamento da UML 1.1 era melhorar a clareza da semântica da UML 1.0 e incorporar as contribuições dos novos parceiros. Foi submetido ao OMG para consideração e adotado no outono de 1997.1 e aprimorado de 1.1 para 1.5 e, posteriormente, para UML 2.1 de 01 para 06 (agora a versão atual da UML é 2.5).

4. POR QUE UML?

À medida que o valor estratégico do software aumenta para muitas empresas, a indústria busca técnicas para automatizar a produção de software, melhorar a qualidade e reduzir o custo e o tempo de colocação no mercado. Essas técnicas incluem tecnologia de componentes, programação visual, padrões e estruturas. As empresas também buscam técnicas para gerenciar a complexidade dos sistemas à medida que aumentam em escopo e escala. Em particular, eles reconhecem a necessidade de resolver problemas recorrentes, como distribuição física, simultaneidade, replicação, segurança, balanceamento de carga e tolerância a falhas. Além disso, o desenvolvimento para a internet, embora torne algumas coisas mais simples, exacerbou esses problemas de arquitetura. A Unified Modeling Language (UML) foi projetada para responder a essas necessidades.

Os principais objetivos no design da UML são:

1. Fornece aos usuários uma linguagem de modelagem visual expressiva e pronta para usar, para que eles possam desenvolver e trocar modelos significativos.
2. Fornece mecanismos de extensibilidade e especialização para estender os conceitos principais.
3. Fornece uma base formal para a compreensão da linguagem de modelagem
4. Incentive o crescimento do mercado de ferramentas OO.
5. Suporte a conceitos de desenvolvimento de nível superior, como colaborações, estruturas, padrões e componentes.
6. Integre as melhores práticas.

A primeira coisa a notar sobre a UML é que existem muitos diagramas (modelos) diferentes com os quais se acostumar. A razão para isso é que é possível olhar para um sistema de muitos pontos de vista diferentes.

As pessoas estão interessadas em diferentes aspectos do sistema e cada uma delas requer um nível diferente de detalhe. Por exemplo, um programador precisa entender o design do sistema e ser capaz de converter o design para um código de baixo nível. A UML tenta fornecer uma linguagem tão expressiva que todos os interessados no sistema possam se beneficiar de pelo menos um diagrama UML.

5. PROJETO DE ESTÁGIO – UML

A proposta como projeto de estágio é: a faculdade tem um projeto para fazer uma avaliação global, onde os alunos de várias faculdades da rede Laureate se inscrevem a partir de um site X. Após a inscrição, o site irá gerar relatórios dos alunos inscritos, faculdades que participam, entre outros.

A ideia é desenhar os casos de uso e esboçar os diagramas de classe desse sistema. Onde os dados de entrada e saída são:

1. Entradas: cadastro do aluno e inscrição em um evento e formação de grupos.
2. Saídas: os administradores podem pedir relatórios dos inscritos e grupos. Os alunos podem verificar seus dados de inscrição

6. DIAGRAMA DE CASOS DE USO

Um diagrama de caso de uso UML é a forma primária de requisitos de sistema / software de um novo programa de software subdesenvolvido. Os casos de uso especificam o comportamento esperado (o quê), e não o método exato de fazê-lo acontecer (como). Os casos de uso, uma vez especificados, podem ser denotados como representação textual e visual. Um conceito chave da modelagem de caso de uso é que ela nos ajuda a projetar um sistema da perspectiva do usuário final. É uma técnica eficaz para comunicar o comportamento do sistema nos termos do usuário, especificando todo o comportamento do sistema visível externamente.

(Tutorials Point 2021) O diagrama de caso de uso é usado para reunir os requisitos de um sistema, incluindo influências internas e externas. Assim, quando um sistema é analisado para reunir suas funcionalidades, os casos de uso são preparados e os atores são identificados. Quando a tarefa inicial é concluída, os diagramas de caso de uso são modelados para apresentar a visão externa.

Em resumo, os objetivos dos diagramas de caso de uso podem ser considerados os seguintes:

1. Usado para reunir os requisitos de um sistema.
2. Usado para obter uma visão externa de um sistema.
3. Identifique os fatores externos e internos que influenciam o sistema.
4. Mostrar a interação entre os requisitos e atores.

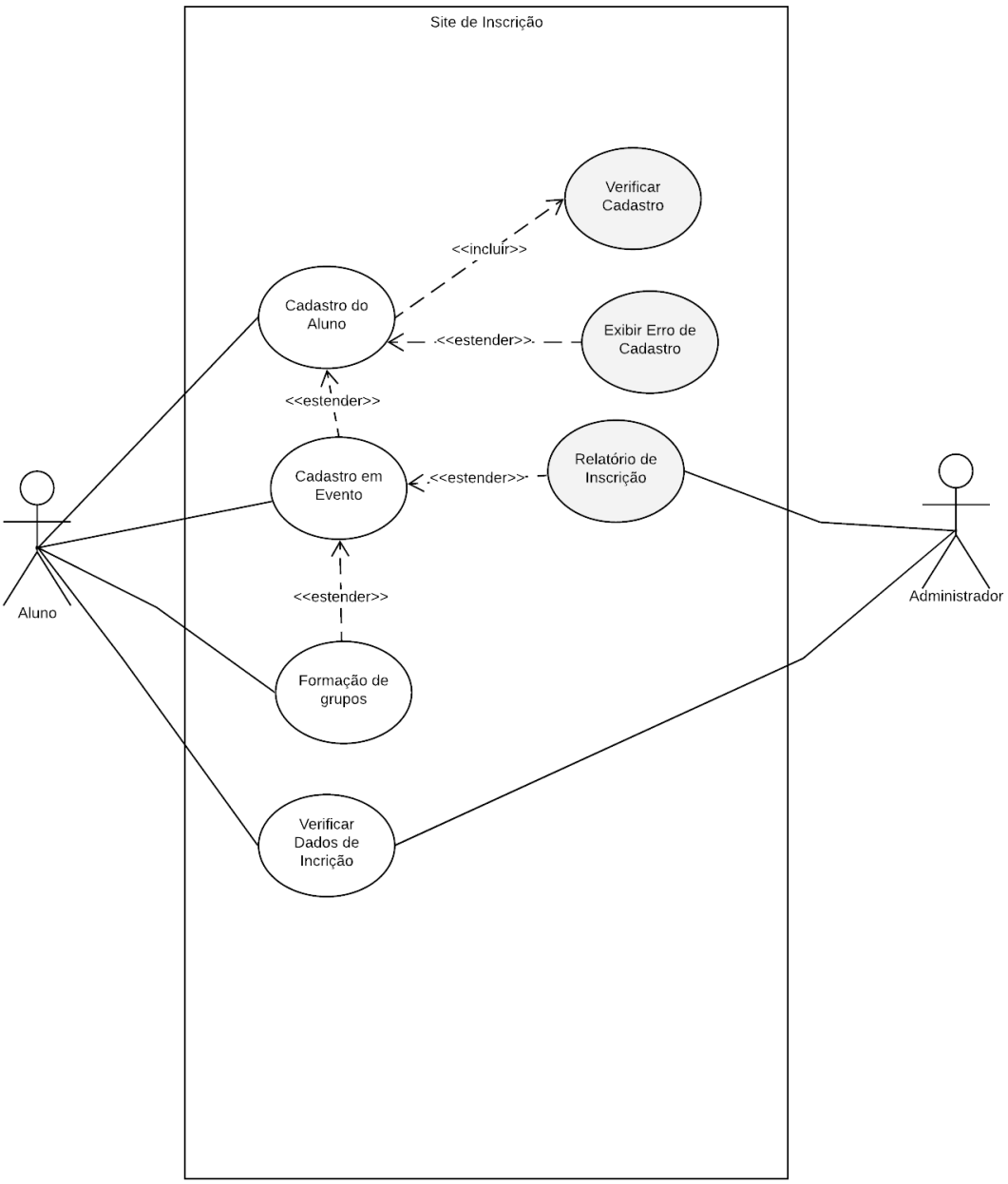
Os diagramas de caso de uso são considerados para análise de requisitos de alto nível de um sistema. Quando os requisitos de um sistema são analisados, as funcionalidades são capturadas nos casos de uso.

Podemos dizer que os casos de uso nada mais são do que as funcionalidades do sistema escritas de forma organizada. A segunda coisa que é relevante para os casos de uso são os atores. Os atores podem ser definidos como algo que interage com o sistema.

Os atores podem ser um usuário humano, alguns aplicativos internos ou podem ser alguns aplicativos externos. Quando planejamos desenhar um diagrama de caso de uso, devemos ter os seguintes itens identificados:

1. Funcionalidades a serem representadas como caso de uso
2. Atores
3. Relacionamentos entre os casos de uso e atores.

Figura 1 - Diagrama de Casos de Uso



No diagrama acima podemos analisar o sistema de inscrição pelo site. Os atores do diagrama estão na parte externa da figura, o aluno e administrador. O aluno é o ator primário, ou seja, o sistema só irá funcionar quando o aluno iniciar a interação, e o administrador é o ator secundário, pois ele irá reagir a essa interação do aluno. O caso de uso representa uma ação que realiza uma tarefa dentro do site, por isso temos essas figuras ovais com um texto dentro representando cada ação que o aluno pode tomar. O aluno irá fazer o cadastro no site e o sistema vai analisar o cadastro, se tiver algo errado deve mostrar um erro para o usuário, por isso temos a seta

“<<estender>>” na figura “Exibir Erro de Cadastro”. Isso é chamado de caso de uso estendido, pois depois que o caso de uso base for realizado (Cadastro do Aluno), o caso de uso estendido pode ocorrer. Também podemos notar a seta “<<incluir>>”, essa seta aponta para um caso de uso incluído, ou seja, o caso de uso base requer o caso de uso incluído para ser completo, por isso temos a figura “Verificar Cadastro”. Depois da verificação de cadastro, o aluno irá se inscrever no evento e vai acontecer a formação dos grupos, gerando a figura “Relatório de Inscrição”, conectada a “Cadastro em Evento” de forma estendida, que pode ser acessada pelo administrador. E por fim, o aluno pode verificar seus dados de inscrição.

7. DIAGRAMA DE CLASSES

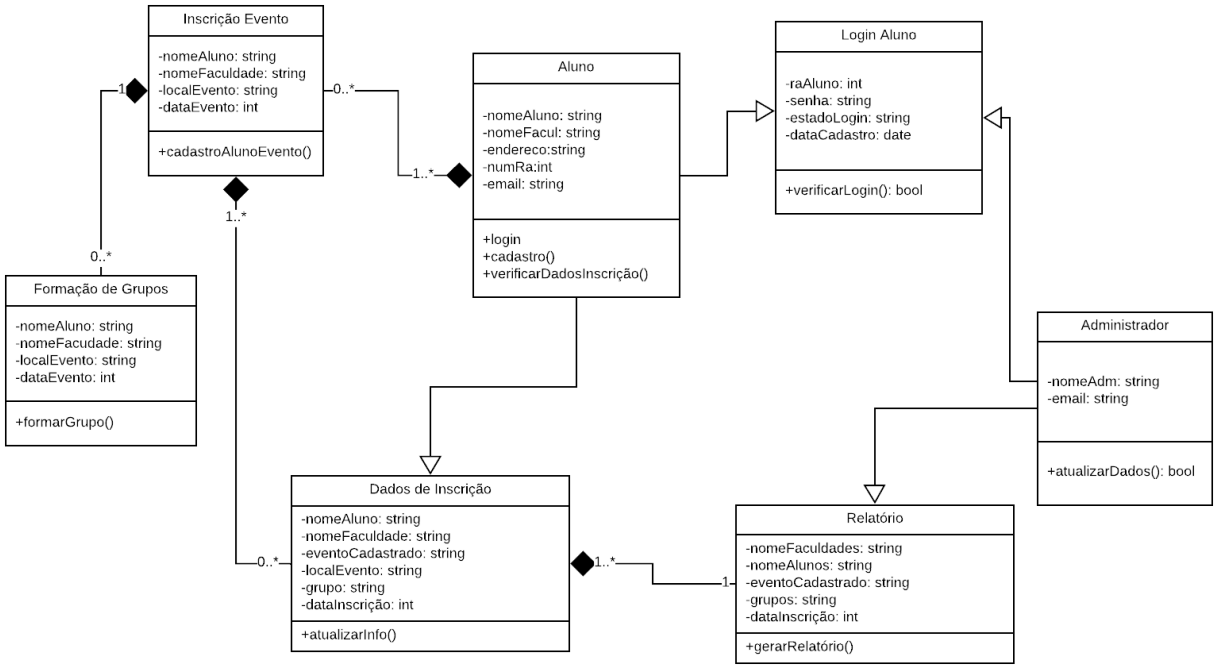
O diagrama de classes é um diagrama estático. Ele representa a visão estática de um aplicativo. O diagrama de classes não é usado somente para visualizar, descrever e documentar diferentes partes de um sistema, mas também para começar a construir o código executável de um aplicativo de software.

(Tutorials Point 2021) Os diagramas UML, como o diagrama de atividades e o diagrama de sequência, por exemplo, podem fornecer apenas o fluxo de sequência do aplicativo, no entanto, o diagrama de classes é um pouco diferente. Ele descreve os atributos e operações de uma classe e também as restrições impostas ao sistema. Os diagramas de classes são amplamente usados na modelagem de sistemas orientados a objetos porque são os únicos diagramas UML, que podem ser mapeados diretamente com linguagens orientadas a objetos.

O objetivo do diagrama de classes pode ser resumido como:

1. Análise e desenho da visão estática de uma aplicação.
2. Descreva as responsabilidades de um sistema.
3. Base para diagramas de componentes e implantação.
4. Engenharia direta e reversa.

Figura 2 - Diagrama de Classes



Na figura acima podemos ter uma visão do sistema de cadastro. Começando pela classe primária “Login Aluno”, recebendo seus atributos privados (-) e com o método pra verificar o login. Esses dados são passados para a classe “Administrador” e “Aluno” por uma relação de herança (seta vazia), ou seja, o administrador e a classe aluno herdam todos os atributos da classe “Login Aluno”. A classe “Aluno” representa a conta do aluno, logo depois que o aluno realizar o cadastro e fazer o login no site, ele poderá se cadastrar em algum evento com a classe “Inscrição Evento”, a classe possui atributos privados e seu método para cadastrar o aluno no evento, ela tem uma relação composta com o cadastro do aluno, por isso está conectada com uma seta preenchida, isso significa que sem a classe “Aluno” a classe “Inscrição Evento” não irá existir, em seguida tem a classe “Formação de Grupos” em um relacionamento composto com a classe “Inscrição Evento” representando a formação de grupos. Também é possível notar a multiplicidade (marcada pelos números nas setas), por exemplo, o aluno pode se inscrever em nenhum evento ou então em vários outros eventos, e o evento pode receber uma inscrição ou várias inscrições de alunos. O diagrama também possui a classe “Dados de Inscrição” conectada por uma seta de relacionamento composto. Os dados da classe “Dados de Inscrição” são passados em forma de herança para a classe “Alunos”, para que os alunos possam verificar os

seus dados de inscrição. E por último a classe “Relatório”, ela está conectada em forma composta com a classe “Dados de Inscrição” pois só existira se existir algum dado de inscrição, e os dados dessa classe são passados em forma de herança para os administradores.

8. BIBLIOGRAFIA

Andrade, A. P. (14 de 09 de 2019). *O que é UML?* Fonte: Traina Web: https://www.treinaweb.com.br/blog/o-que-e-uml/

Leandro. (13 de 04 de 2021). *O que é UML e Diagramas de Caso de Uso: Introdução Prática à UML*. Fonte: DevMedia: https://www.devmedia.com.br/o-que-e-uml-e-diagramas-de-caso-de-uso-introducao-pratica-a-uml/23408

Noleto, C. (27 de 06 de 2020). *UML: o que é e para que serve essa linguagem de notação?* Fonte: Betrybe: https://blog.betrybe.com/tecnologia/uml/

Point, T. (03 de 04 de 2021). *UML - Diagrama de Casos de Uso*. Fonte: Tutorials Point: https://www.tutorialspoint.com/uml/uml_use_case_diagram.htm

Point, T. (13 de 04 de 2021). *UML - Diagrama de Classes*. Fonte: Tutorials Point: https://www.tutorialspoint.com/uml/uml_class_diagram.htm

Ventura, P. (03 de 01 de 2019). *O que é UML (Unified Modeling Language)*. Fonte: Até o momento: https://www.ateomomento.com.br/diagramas-uml/

What is Unified Modeling Language (UML)? . (13 de 04 de 2021). Fonte: Visual Paradigm: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/