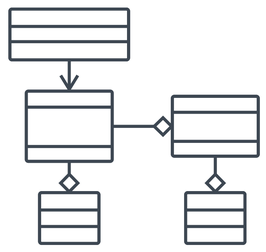


O que é um diagrama UML?

Por que usar um diagrama UML?

Quero aprender mais sobre diagramas de caso de uso pois são novidade para mim. ↓

Eu quero criar meu próprio diagrama de caso de uso no Lucidchart.

Eu quero criar um diagrama de caso de uso a partir de um modelo do Lucidchart.

Conteúdo

O que é UML?

A UML e seu papel na modelagem e no design orientados a objetos

A história e as origens da UML

Padrões OMG

O objetivo da UML de acordo com a OMG

UML e modelagem de dados

Atualizações em UML 2.0

Glossário de termos UML

Conceitos de modelagem especificados pela UML

Conceitos orientados a objetos em UML

Tipos de diagramas UML

Como criar um diagrama UML: tutoriais e exemplos

O Lucidchart facilita a criação de diagramas UML

Uma imagem vale mais do que mil palavras. Por este motivo, a diagramação em Linguagem de modelagem unificada (UML) foi criada: para estabelecer uma linguagem visual comum no complexo mundo do desenvolvimento de software, que também poderia ser compreendida por usuários do mundo dos negócios e qualquer pessoa que queira entender mais sobre um sistema. Aprenda mais sobre os fundamentos de diagramas UML, juntamente com sua origem, seus usos, conceitos, tipos e orientações sobre como desenhá-los usando a nossa ferramenta de diagramas UML.

Leitura de 17 minuto(s)

Quer criar um diagrama UML próprio? Experimente o Lucidchart. É rápido, fácil e completamente gratuito.

Criar um diagrama UML

O que é UML?

A Linguagem de modelagem unificada (UML) foi criada para estabelecer uma linguagem de modelagem visual comum, semanticamente e sintaticamente rica, para arquitetura, design e implementação de sistemas de software complexos, tanto estruturalmente quanto para comportamentos. Além do desenvolvimento de software, a UML tem aplicações em fluxos do processo na fabricação.

É análoga aos modelos utilizados em outros campos, e é composta por diferentes tipos de diagramas. De modo geral, diagramas UML descrevem o limite, a estrutura e o comportamento do sistema e os objetos nele contidos.

A UML não é uma linguagem de programação, mas existem ferramentas que podem ser usadas para gerar código em várias linguagens por meio de diagramas UML. A UML tem uma relação direta com a análise e o design orientados a objetos.

A UML e seu papel na modelagem e no design orientados a objetos

Existem muitos paradigmas ou modelos de resolução de problemas na ciência da computação, que é o estudo de algoritmos e dados. Há quatro categorias modelo de resolução de problemas: linguagens imperativas, funcionais, declarativas e orientadas a objetos (POO). Em linguagens orientadas a objetos, os algoritmos são expressos através da definição de 'objetos', e por meio da interação dos objetos uns com os outros. Esses objetos são coisas a serem manipulados, e eles existem no mundo real. Eles podem ser edifícios, widgets em um desktop ou seres humanos.

Linguagens orientadas a objetos dominam o mundo da programação porque elas modelam objetos do mundo real. A UML é uma combinação de várias notações orientadas a objetos: design orientado a objetos, técnica de modelagem de objetos e engenharia de software orientada a objetos.

A UML usa os pontos fortes destas três abordagens para apresentar uma metodologia mais consistente e mais fácil de usar. A UML representa as melhores práticas para desenvolver e documentar aspectos diferentes da modelagem de software e sistemas de negócios.

A história e as origens da UML

'Os Três Amigos' da engenharia de software, como eram conhecidos, desenvolveram ainda outras metodologias. Uniram-se para proporcionar maior clareza aos programadores por meio da criação de novos padrões. A colaboração entre Grady, Booch e Rumbaugh fortaleceu os três métodos e melhorou o produto final.

Em 1996, os grandes esforços destes pensadores resultaram no lançamento dos documentos UML 0.9 e 0.91. Logo ficou claro que muitas organizações, incluindo Microsoft, Oracle e IBM, passaram a considerar a UML como algo crucial para seu próprio desenvolvimento de negócios. Juntamente com muitos outros indivíduos e empresas, as organizações estabeleceram recursos que poderiam desenvolver uma linguagem de modelagem plena. Em 1999, Os Três Amigos publicaram "UML: guia do usuário", e uma atualização que inclui informações sobre UML 2.0, em sua segunda edição, de 2005.

Padrões OMG

Segundo o site, The **Object Management Group®** (OMG®) (Grupo de gerenciamento de objetos, em tradução livre) é uma organização internacional voltada para aprovações de padrões tecnológicos, aberta a membros, sem fins lucrativos e fundada em 1989. O padrões OMG são implementados por fornecedores, utilizadores finais, instituições acadêmicas e agências governamentais. Forças-tarefas OMG desenvolvem padrões de integração empresarial para muitas tecnologias e um grande número de indústrias. Os padrões de modelagem OMG, incluindo UML e Model Driven Architecture® (MDA®), possibilitam uma criação de design visualmente poderoso e a execução e manutenção de software, entre outros processos.

A OMG supervisiona a definição e manutenção de especificações UML. Esta supervisão permite aos engenheiros e programadores usarem uma única linguagem para diversas finalidades durante todas as fases do ciclo de vida do software e para todos os tamanhos de sistemas.

O objetivo da UML de acordo com a OMG

O propósito da UML de acordo com a OMG:

- Fornecer a arquitetos de sistemas, engenheiros de software e desenvolvedores de software ferramentas de análise, design e implementação para sistemas baseados em software, bem como para a modelagem de processos de negócios e similares.
- Desenvolver as condições gerais da indústria ao permitir a interoperabilidade de ferramentas de modelagem visual de objetos. No entanto, para permitir a troca significativa de informações de modelos entre as ferramentas, é necessário um acordo sobre semântica e notação.

A UML atende aos seguintes requisitos:

- Estabelecer uma definição formal de um metamodelo baseado em meta-objetos (MOF, em inglês) comuns que especifica a sintaxe abstrata da UML. A sintaxe abstrata define o conjunto de conceitos de modelagem UML, seus atributos e relacionamentos, bem como as regras para combinar estes conceitos para construir modelos UML parciais ou completos.
- Fornecer uma explicação detalhada da semântica de cada conceito de modelagem UML. A semântica define, de forma independente da tecnologia, como os conceitos UML devem ser realizados por computadores.
- Especificar os elementos de notação legíveis para humanos para representar os conceitos de modelagem UML individuais, bem como regras para combiná-los em uma variedade de tipos de diagramas correspondentes a diferentes aspectos dos sistemas modelados.
- Definir maneiras pelas quais ferramentas UML podem entrar em conformidade com esta especificação. Isto é suportado (em uma especificação separada) por uma especificação baseada em XML de formatos de modelos de intercâmbio correspondentes (XMI) que devem ser realizados por ferramentas compatíveis.

UML e modelagem de dados

A UML é bastante usada pelos programadores, mas não é geralmente empregada por desenvolvedores de banco de dados. Um motivo é simplesmente que os criadores da UML não focaram banco de dados. Apesar disso, a UML é eficaz para níveis altos de modelagem de dados conceituais, e pode ser usada em diferentes tipos de diagramas UML. Você pode encontrar mais informações sobre como criar camadas de um modelo de classe orientado a objetos em um banco de dados relacional neste artigo sobre **Modelagem de banco de dados em UML.**

Quer criar um diagrama UML próprio? Experimente o Lucidchart. É rápido, fácil e completamente gratuito.

Criar um diagrama UML

Atualizações em UML 2.0

A UML é aprimorada de forma contínua. UML 2.0 estende as especificações UML para abranger mais aspectos do desenvolvimento, incluindo Agile. O objetivo foi reestruturar e aperfeiçoar a UML para que a usabilidade, implementação e adaptação fossem simplificadas. Seguem algumas das atualizações aos diagramas UML:

- Maior integração entre modelos estruturais e comportamentais.
- Capacidade de definir hierarquia e dividir um sistema de software em componentes e subcomponentes.
- UML 2.0 aumenta o número de diagramas de 9 para 13.

Glossário de termos UML

Familiarize-se com o vocabulário UML com esta lista obtida do documento UML 2.4.1 destinado a ajudar não membros da OMG a entender termos frequentemente usados.

- **Conformidade de sintaxe abstrata** Usuários podem mover modelos através de diferentes ferramentas, mesmo se usam diferentes notações
- **Common Warehouse Metamodel (CWM)** O Metamodelo de armazenamento comum, em tradução livre, é uma interface padrão usada para permitir o intercâmbio de metadados de armazenamento e de inteligência de negócios entre ferramentas de armazenamento, plataformas de armazenamento e repositórios de armazenamento de metadados em ambientes heterogêneos e distribuídos
- **Conformidade de sintaxe concreta** Por meio de diferentes ferramentas, usuários podem continuar a usar uma notação familiar a eles
- **Core** No contexto da UML, o core geralmente se refere ao “Pacote core”, que é um metamodelo completo especialmente projetado para uma alta capacidade de reutilização
- **Unidade de linguagem** Consiste em uma coleção de conceitos de modelagem fortemente acoplados que oferece aos usuários o poder de representar aspectos do sistema em estudo de acordo com um determinado paradigma ou formalismo
- **Nível 0 (N0)** Primeiro nível de conformidade da infraestrutura UML – uma única unidade de linguagem que ajuda a modelar os tipos de estruturas baseadas em classes encontradas nas linguagens mais usadas de programação orientada a objetos
- **Meta-objetos (MOF)** Uma especificação de modelagem OMG que fornece a base para a definição de metamodelos na família de linguagens MDA da OMG
- **Metamodelo** Define a linguagem e os processos dos quais se cria um modelo
- **Construções de metamodelos (LM)** Segundo nível de conformidade na infraestrutura UML – uma unidade de linguagem extra para estruturas baseadas em classes mais avançadas utilizadas para a construção de metamodelos (usando construções de meta-objetos, CMOF), como a própria UML. A UML tem apenas dois níveis de conformidade
- **Modelo Driven Architecture (MDA)** Uma abordagem e um plano para atingir um conjunto coeso de especificações de tecnologia baseadas em modelos
- **Linguagem para especificação de restrições em objetos** Object Constraint Language (OCL), em inglês, é a linguagem declarativa para descrever as regras que se aplicam à Linguagem de modelagem unificada A OCL suplementa a UML ao fornecer termos e símbolos de fluxogramas mais precisos que a linguagem natural, e menos difíceis de dominar que a matemática
- **Object Management Group (OMG)** É uma organização voltada para especificações para a indústria computacional, sem fins lucrativos, cujos membros definem e mantêm a especificação UML
- **UML 1** Primeira versão da Linguagem de modelagem unificada
- **Linguagem de modelagem unificada (UML)** Uma linguagem visual para especificar, criar e documentar os artefatos de sistemas
- **XMI** Uma especificação baseada em XML de formatos de modelos de intercâmbio correspondentes

Veja o documento MOF completo

Baixe, por completo, o documento de infraestrutura UML 2.4.1.

Conceitos de modelagem especificados pela UML

O desenvolvimento de sistemas centra-se em três modelos gerais e diferentes do sistema:

- **Funcional:** são diagramas de caso de uso, e descrevem a funcionalidade do sistema a partir do ponto de vista do usuário.
- **Objeto:** são diagramas de classes, e descrevem a estrutura do sistema em termos de objetos, atributos, associações e operações.
- **Dinâmico:** diagramas de interação, diagramas de máquinas de estados e diagramas de atividade, e são usados para descrever o comportamento interno do sistema.

Estes modelos de sistemas são visualizados por meio de dois tipos diferentes de diagramas: estrutural e comportamental.

Conceitos orientados a objetos em UML

Os objetos em UML são entidades do mundo real que existem ao nosso redor. No desenvolvimento de software, objetos podem ser usados para descrever, ou modelar, o sistema que está sendo criado em termos relevantes ao domínio. Objetos também permitem decompor sistemas complexos em componentes compreensíveis, em que uma peça de cada vez pode ser criada.

Seguem alguns conceitos fundamentais de um mundo orientado a objetos:

- **Objetos** Representam uma entidade e um componente básico essencial.
- **Classe** Modelo de um objeto.
- **Abstração** Comportamento de uma entidade do mundo real.
- **Encapsulamento** Mecanismo de vincular os dados e escondê-los do mundo exterior.
- **Herança** Mecanismo de criar novas classes a partir de uma existente.
- **Polimorfismo** Define o mecanismo em diferentes formas.

Tipos de diagramas UML

A UML usa elementos e associa-os de diferentes maneiras para formar diagramas que representam aspectos estáticos ou estruturais de um sistema, além de formar diagramas comportamentais, que registram os aspectos dinâmicos de um sistema.

Diagramas UML estruturais

- **Diagrama de classes** É o diagrama UML mais usado, e a principal base de qualquer solução orientada a objetos. Classes dentro de um sistema, atributos e operações, e a relação entre cada classe. Classes são agrupadas para criar diagramas de classes quando há uma diagramação de grandes sistemas.
- **Diagrama de componentes** Exibe a relação estrutural de elementos do sistema de software, na maioria das vezes utilizado quando se trabalha com sistemas complexos com múltiplos componentes. Componentes se comunicam por meio de interfaces.
- **Diagrama de estrutura composta** Diagramas de estrutura composta são utilizados para mostrar a estrutura interna de uma classe.
- **Diagrama de implementação** Ilustra o hardware do sistema e seu software. É útil quando uma solução de software é implantada em diversas máquinas com configurações únicas.
- **Diagrama de objetos** Mostra a relação entre objetos usando exemplos do mundo real e retrata um sistema em um determinado momento. Como os dados estão disponíveis dentro de objetos, eles podem ser utilizados para esclarecer as relações entre objetos.
- **Diagrama de pacotes** Existem dois tipos especiais de dependências definidas entre pacotes: a importação do pacote e a mesclagem do pacote. Para revelar a arquitetura, os pacotes representam os diferentes níveis de um sistema. Dependências de pacotes podem ser marcadas para mostrar o mecanismo de comunicação entre os níveis.

Diagramas UML comportamentais

- **Diagramas de atividade** Fluxos de trabalho de negócios ou operacionais representados graficamente para exibir a atividade de qualquer parte ou componente do sistema. Diagramas de atividade são usados como alternativa aos diagramas de máquina de estados.
- **Diagrama de comunicação** Semelhante a diagramas de sequência, no entanto foca mensagens transmitidas entre objetos. A mesma informação pode ser representada usando um diagrama de sequência e outros objetos.
- **Diagrama da visão geral da interação** Há sete tipos de diagramas de interação, e este diagrama exibe a sequência em que eles atuam.
- **Diagrama de sequência** Mostra como objetos interagem entre si, e a ordem de ocorrência. Representam interações para um determinado cenário.
- **Diagrama de máquina de estados** Semelhante a diagramas de atividade, eles descrevem o comportamento de objetos que se comportam de maneiras diferentes em seu estado atual.
- **Diagrama de tempo** Assim como os diagramas de sequência, representa o comportamento de objetos em um determinado período de tempo. Se houver um único objeto, o diagrama é simples. Se houver mais de um objeto, as interações dos objetos são exibidas durante este período de tempo determinado.
- **Diagrama de caso de uso** Representa uma determinada funcionalidade de um sistema, e foi criado para ilustrar a forma como as funcionalidades se relacionam e seus controladores internos e externos (atores).

Como criar um diagrama UML: tutoriais e exemplos

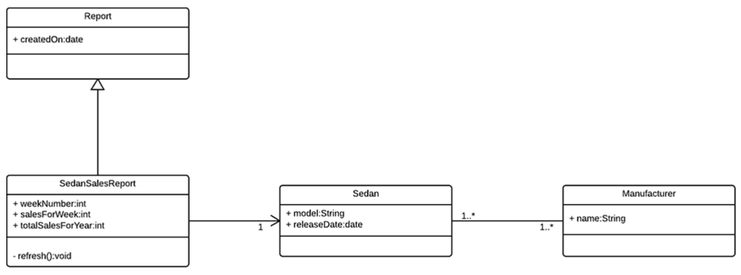
Para criar diferentes tipos de diagramas UML, experimente um destes tutoriais (ou todos) para orientá-lo no processo de desenhar diagramas estruturais e comportamentais.

Exemplos de tutoriais de diagramas estruturais

DIAGRAMAS DE CLASSE

Diagramas de classe representam as estruturas estáticas de um sistema, incluindo suas classes, atributos, operações e objetos. Um diagrama de classe exibe dados organizacionais ou computacionais na forma de classes lógicas e de implementação, respectivamente. Pode haver sobreposição entre estes dois grupos.

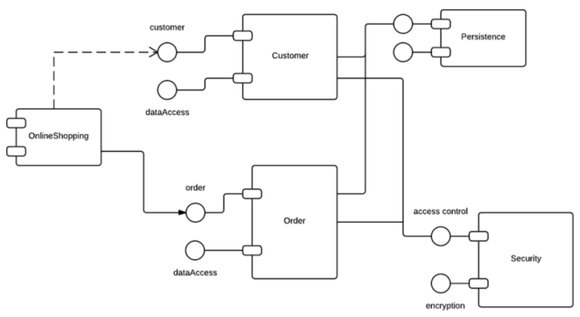
1. Classes são representadas por uma forma retangular, dividida em três partes. A parte superior exibe o nome da classe, enquanto a parte do meio contém os atributos da classe. A parte inferior representa as operações da classe (também conhecida como métodos).
2. Adicione formas de classes ao seu diagrama de classe para modelar a relação entre os objetos. Pode ser necessário adicionar subclasses.
3. Use linhas para representar associações, heranças, multiplicidades e outras relações entre as classes e subclasses. Seu estilo de notação preferido ditará a notação destas linhas.



DIAGRAMAS DE COMPONENTES

Diagramas de componentes mostram como os componentes são combinados para formar componentes maiores ou sistemas de software. Estes diagramas são destinados a modelar as dependências de cada componente do sistema. Um componente é necessário para executar uma função de estereótipo. Um componente de estereótipo pode conter executáveis, documentos, tabelas de banco de dados, arquivos ou arquivos de biblioteca.

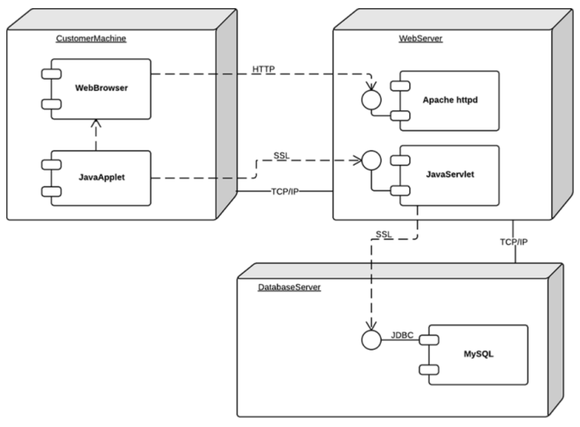
1. Representam um componente com uma forma de retângulo. Deve conter dois pequenos retângulos no lado, ou apresentar um ícone com esta forma.
2. Adicionam linhas entre as formas de componentes para representar as relações relevantes.



DIAGRAMAS DE IMPLEMENTAÇÃO

Um diagrama de implementação modela a implementação física e estrutura de componentes de hardware. Diagramas de implementação demonstram onde e como os componentes de um sistema vão operar uns com os outros.

1. Quando for desenhar um diagrama de implementação, use a mesma notação que você usa para um diagrama de componentes.
2. Use um cubo 3-D para modelar um nó (que representa uma máquina física ou virtual).
3. Rotule o nó no mesmo estilo que foi usado para diagramas de sequência. Adicione outros nós, conforme necessário, e, em seguida, conecte por meio de linhas.

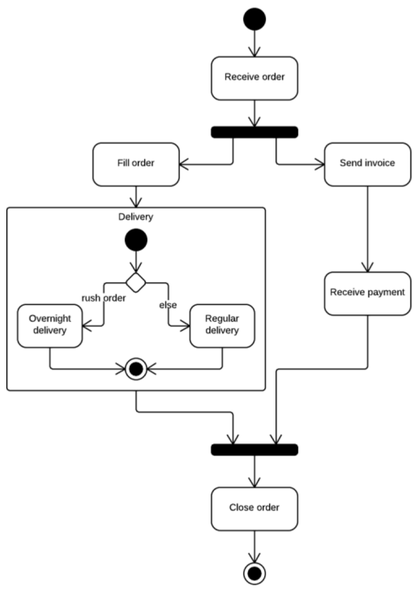


Exemplos de tutoriais de diagramas comportamentais

DIAGRAMAS DE ATIVIDADE

Diagramas de atividade exibem o fluxo processual do controle entre objetos de classe, juntamente com processos organizacionais, como fluxos de trabalho empresariais. Estes diagramas são feitos de formas especializadas, e então conectados com setas. A notação definida para diagramas de atividade é semelhante aos dos diagramas de estados.

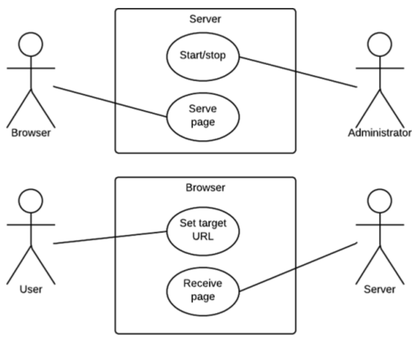
1. Comece o seu diagrama de atividade com um círculo sólido.
2. Conecte o círculo à primeira atividade, modelada por um retângulo de cantos arredondados.
3. Agora, conecte cada atividade a outras atividades, com linhas que demonstram o fluxo passo a passo de todo o processo.
4. Você também pode tentar usar raias para representar os objetos que executam cada atividade.



DIAGRAMAS DE CASO DE USO

Um caso de uso é uma lista de passos que definem a interação entre um ator (um ser humano que interage com o sistema, ou com um sistema externo) e o próprio sistema. Diagramas de caso de uso descrevem as especificações de um caso de uso e modelam as unidades funcionais de um sistema. Estes diagramas ajudam equipes de desenvolvimento a entender as necessidades de seu sistema, incluindo o papel da interação humana no seu interior e as diferenças entre vários casos de uso. Um diagrama de caso de uso pode exibir todos os casos de uso do sistema, ou apenas um grupo de casos de uso com funcionalidade semelhante.

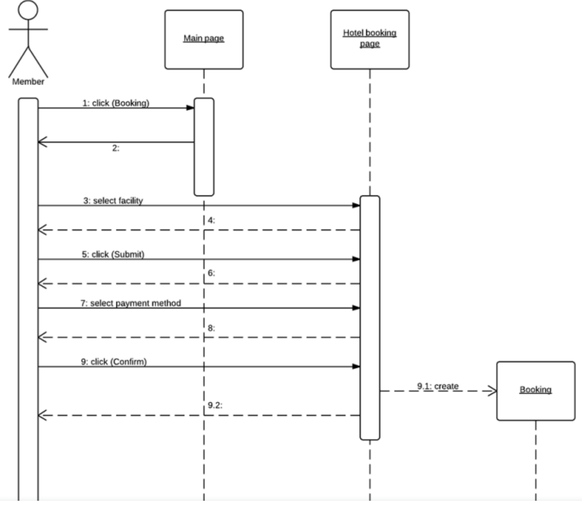
1. Para iniciar um diagrama de caso de uso, adicione uma forma oval ao centro do desenho.
2. Digite o nome do caso de uso dentro do oval.
3. Represente atores com um boneco palito próximo ao diagrama, em seguida, use linhas para modelar as relações entre atores e casos de uso.



DIAGRAMAS DE SEQUÊNCIA

Diagramas de sequência, também conhecidos como diagramas de eventos ou cenários de eventos, ilustram a forma como os processos interagem uns com os outros, exibindo chamadas feitas entre objetos diferentes em uma sequência. Estes diagramas contêm duas dimensões: vertical e horizontal. As linhas verticais mostram a sequência de mensagens e chamadas em ordem cronológica, e os elementos horizontais mostram instâncias de objetos onde as mensagens são retransmitidas.

1. Para criar um diagrama de sequência, escreva o nome da instância de classe e o nome da classe em uma caixa retangular.
2. Desenhe linhas entre as instâncias de classe para representar quem envia e quem recebe as mensagens.
3. Use pontas de setas sólidas para simbolizar mensagens síncronas, pontas de setas abertas para mensagens assíncronas e linhas tracejadas para mensagens de resposta.



Lucidchart

O Lucidchart facilita a criação de diagramas UML

Você pode começar a diagramar com UML agora mesmo com o Lucidchart. Tornamos a diagramação simples, eficiente e até mesmo divertida.

- **Fácil de usar** Nós sabemos muito bem que você sabe criar um diagrama UML, por isso queremos facilitar o seu trabalho o máximo possível. Você economizará tempo com a interface sofisticada e o editor inteligente de arrastar e soltar do Lucidchart.
- **Extensa biblioteca de formas** Desenhe diagramas de estados, diagramas de atividade, diagramas de caso de uso e muito mais. Com uma biblioteca extensa de formas e conectores, você encontrará tudo de que precisa.
- **Integração completa** O Lucidchart faz integração completa com o G Suite. Ao usar o Lucidchart, será possível nos acessar em seu Suite de produtividade do Google, junto com o Gmail e Google Drive. E você poderá ainda usar o mesmo login do Google.
- **Permite a colaboração** Você poderá compartilhar facilmente seu diagrama UML com seus colegas, clientes ou chefe. Seus diagramas podem ser incorporados em uma página da web ou publicados como PDF, e o modo apresentação do Lucidchart deixa seu projeto com uma ótima aparência, pronto para apresentar.
- **Importação/Exportação Visio** É fácil importar e exportar arquivos do Visio, permitindo salvar seus projetos já concluídos. A experiência toda é rápida e ininterrupta.

Recursos úteis

Tudo sobre diagramas de estrutura composta →

O que é um diagrama de sequência UML? →

O que é um diagrama de máquina de estados? →

Tudo sobre diagrams de interação UML →

Tudo sobre diagramas de pacotes UML →

O que é um diagrama de implementação? →

Diagrama de caso de uso UML: O que é, como fazer e exemplos →

O que é um diagrama de objetos? →

O que é diagrama de atividades UML? →

O que é um diagrama de classe UML? →

Diagrama de componentes UML: o que é, como fazer e exemplos →

Software de diagramas UML online →

Quer criar um diagrama UML próprio? Experimente o Lucidchart. É rápido, fácil e completamente gratuito.

Criar um diagrama UML