Fatec Registro

Centro Paula Souza

Gestão Ágil de Projetos

Professor – Ramon Trigo

Objetivos de Aprendizagem

- Compreender e aplicar conceitos, técnicas e ferramentas para revisar e aprimorar os modelos e processos de gestão de projetos de software.
- Empregar no gerenciamento de projetos de software as melhores práticas ágeis.

Ementa

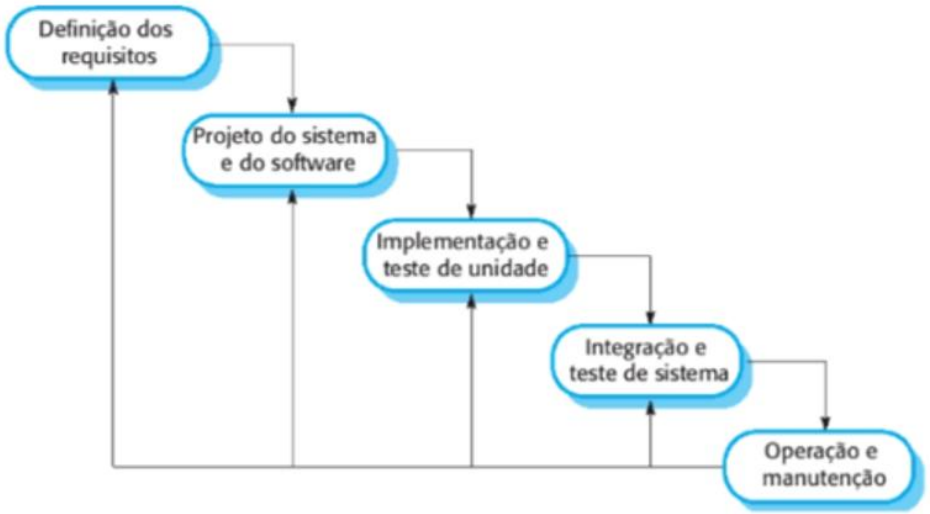
- Visão geral sobre o gerenciamento tradicional de projetos versus metodologia ágil de projetos. Introdução aos métodos ágeis, princípios, valores e filosofia.
- O manifesto Ágil. Principais métodos ágeis.
- Princípios e práticas ágeis para gestão de projetos de desenvolvimento de software. Práticas de gerenciamento ágil de projetos (engajamento das partes interessadas, gerência da equipe, planejamento adaptativo, detecção e resolução de problemas e melhoria contínua).
- Papéis e responsabilidades de equipes ágeis.
- Gestão de Performance e Gestão de Times Ágeis.

Conhecendo a Grade do Curso – 3° Semestre

Período	Sigla	Relação de Componentes	Modalidade	Aulas Semestrais			Total de Aulas Semestrais
				Sala de Aula	Laboratório	Remota –Síncrona	
3° semestre	ILP-037	Técnicas de Programação II	Presencial	-	80	-	80
	ISW-030	Desenvolvimento Web III	Presencial	-	80	-	80
	MAG-004	Álgebra Linear	Presencial	80	-	-	80
	AGO-021	Gestão Ágil de Projetos de Software	Presencial	-	80		80
	IBD-016	Banco de Dados - Não Relacional	Presencial	-	80	-	80
	IHC-004	Interação Humano Computador	Presencial	-	40		40
	ING-085	Inglês I	Presencial	40	-	-	40
	Total de aulas semestrais			120	360	-	480

Conhecendo Métodos Ágeis

Nos anos 1980 e início dos anos 1990. havia uma percepção generalizada de que a maneira mais indicada para obter um software melhor era por meio do planejamento cuidadoso do projeto, da garantia de qualidade formalizada, do uso de métodos de análise e projeto (design) apoiados por ferramentas de software e de processos de desenvolvimento controlados e rigorosos. Essa percepção veio da comunidade de engenharia de software que era responsável por desenvolver sistemas grandes e duradouros.



Introdução ao UML

UML (Unified Modeling Language – linguagem de modelagem unificada) é “uma linguagem-padrão para descrever/documentar projeto de software. A UML pode ser usada para visualizar, especificar, construir e documentar os artefatos de um sistema de software-intensivo” .

Em outras palavras, assim como os arquitetos criam plantas e projetos para ser usados por uma empresa de construção, os arquitetos de software criam diagramas UML para ajudar os desenvolvedores de software a construir o software.

Se você entender o vocabulário da UML (os elementos visuais do diagrama e seus significados), pode facilmente entender e especificar um sistema e explicar o projeto daquele sistema para outros interessado

O que é UML

O grande problema do desenvolvimento de novos sistemas utilizando a orientação a objetos nas fases de análise de requisitos, análise de sistemas e design é que não existe uma notação padronizada e realmente eficaz que abranja qualquer tipo de aplicação que se deseje.

Quando a “Unified Modeling Language” (UML) foi lançada, muitos desenvolvedores da área da orientação a objetos ficaram entusiasmados já que essa padronização proposta pela UML era o tipo de força que eles sempre esperaram.

Por que Modelar um Software / Projeto

Qual a real necessidade de se projetar um casa? Um pedreiro experiente não e capaz de construí-la sem um projeto?

Por que Modelar um Software / Projeto

Para se construir um edifício é necessário, em primeiro lugar, desenvolver um projeto muito bem-elaborado, cujos cálculos têm de estar corretos e precisos.

Necessário fornecer uma estimativa de custos, determinar em quanto tempo a construção estará concluída, especificar a quantidade de material a ser adquirida para a construção escolher o local onde o prédio será erguido.

Grandes projetos não podem ser modelados de cabeça.

Por que Modelar um Software

Um sistema de informação precisa ter uma documentação extremamente detalhada, precisa e atualizada para que possa ser mantido com facilidade, rapidez e correção, sem produzir novos erros ao corrigir os antigos.

Modelar um sistema é uma forma bastante eficiente de documenta-lo, mas a modelagem não serve apenas para isso. A documentação é apenas uma das vantagens fornecidas pela modelagem

Modelo de Software

Um modelo de software captura uma visão de um sistema físico é uma abstração do sistema com um certo proposito, como descrever aspectos estruturais ou comportamentais do software.

Assim um modelo descreve completamente aqueles aspectos do sistema físico que são relevantes ao propósito do modelo, no nível apropriado de detalhe.

Fatec Registro

Centro Paula Souza

Diagramas UML

A UML divide os diagramas em duas categorias: estruturais e comportamentais.

Os diagramas estruturais podem ser utilizados para documentar a organização física, como um objeto se relaciona com outro.

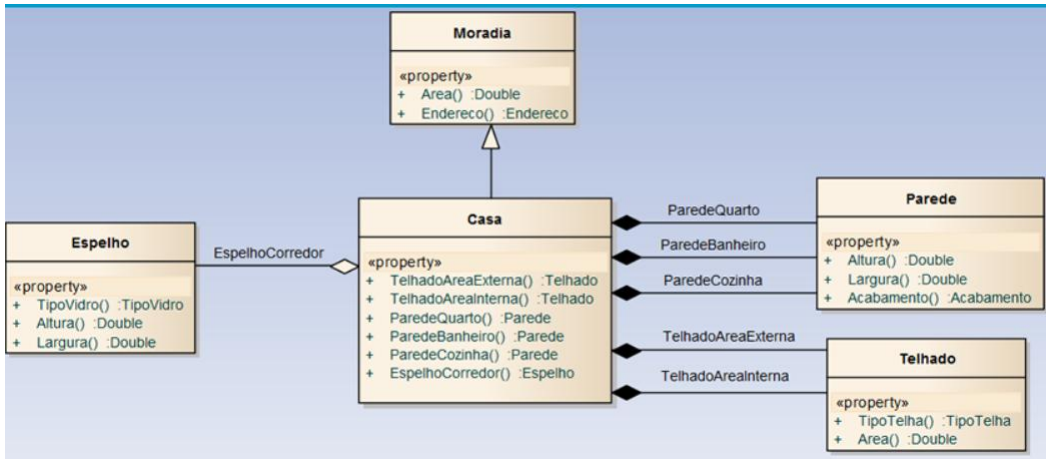
Os diagramas comportamentais tem foco no comportamento dos elementos de um sistema.

Por exemplo, para documentar requisitos, operações e alterações internas de estado para elementos.

Estrutural

Diagrama de Classe: utilizam classes e interfaces para documentar detalhes sobre as entidades que formam o sistema e as relações estáticas entre elas.

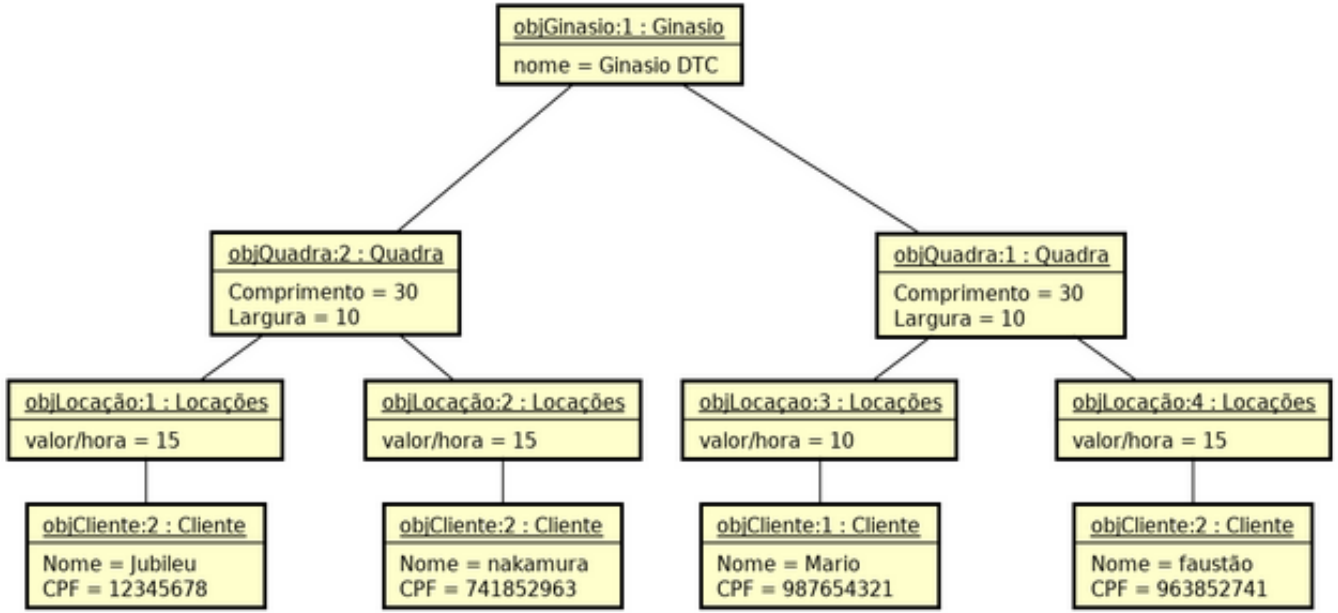
Os diagramas de classe estão entre os mais utilizados e podem variar em detalhes que podem ser depurados e aptos para gerar código fonte, a esboços rápidos em quadros brancos.



Fonte: https://www.ateomomento.com.br/uml-diagrama-de-classes/

Estrutural

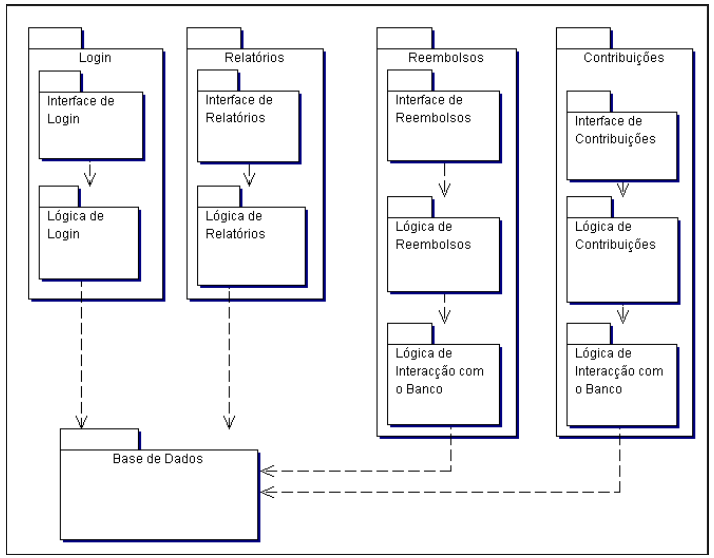
Diagramas de Objeto: utilizam a mesma sintaxe que os diagramas de classes e mostram como as etapas reais de classe são relacionadas num instante especifico. Você pode usar os diagramas de objetos para mostrar as relações dentro do sistema em momento especifico.



Fonte: https://medium.com/documenta%C3%A7ao-uml/introdu%C3%A7%C3%A3o-ao-diagrama-de-objetos-902795d485f8

Estrutural

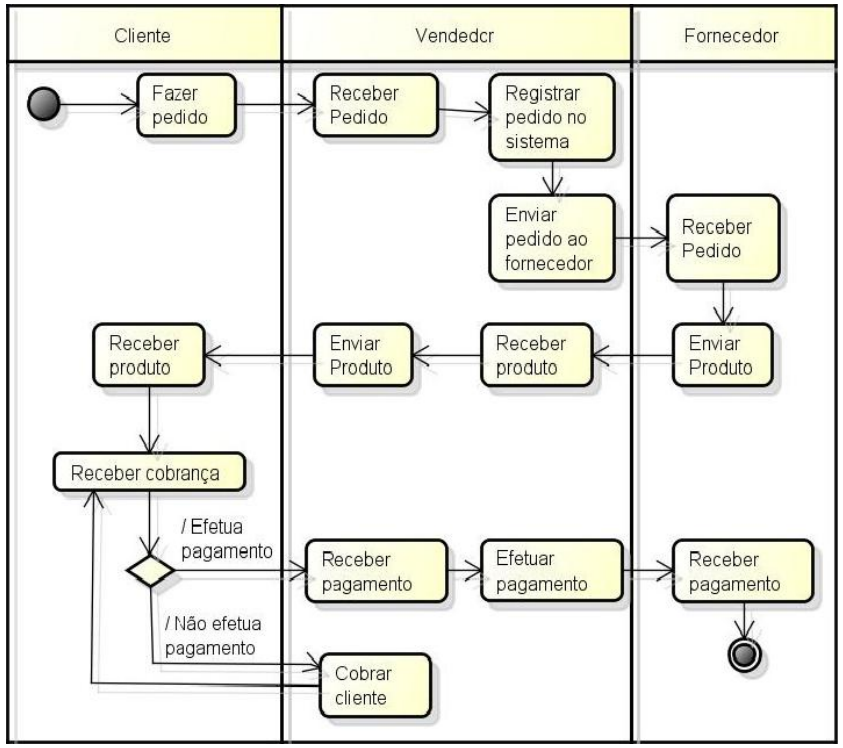
Diagrama de pacote: são realmente tipos especiais de diagramas de classe. Eles utilizam a mesma notação, mas o foco está em como as classes e interfaces são agrupadas.



https://micreiros.com/diagrama-de-pacotes/

Comportamental

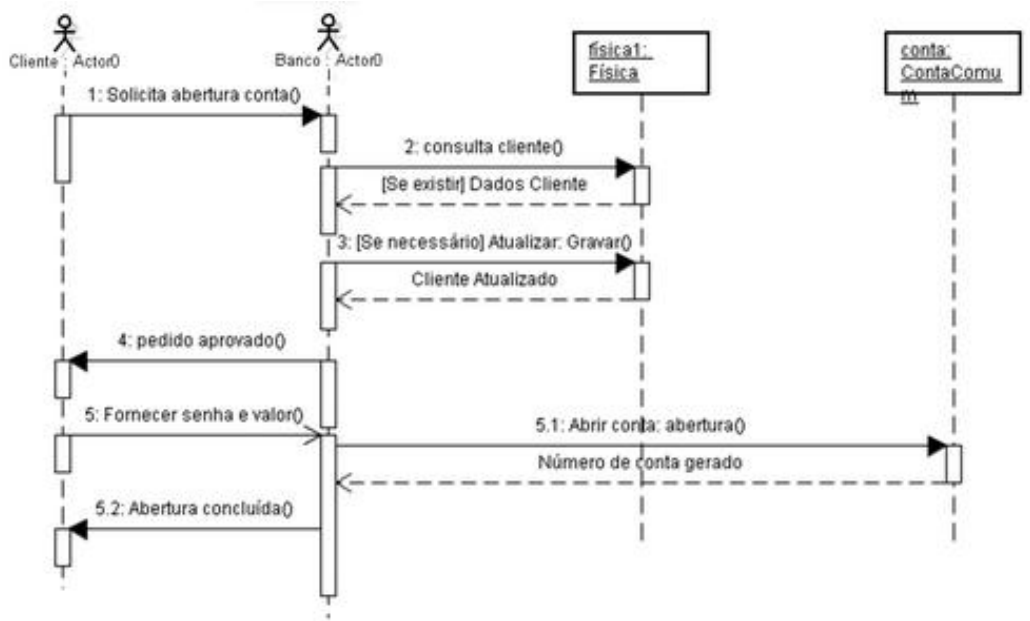
Diagramas de atividade Fluxos de trabalho de negócios ou operacionais representados graficamente para exibir a atividade de qualquer parte ou componente do sistema. Diagramas de atividade são usados como alternativa aos diagramas de máquina de estados.



Fonte: https://www.researchgate.net/profile/Ildeberto-Rodello/publication/280319043/figure/fig2/AS:366637279662081@1464424547133/Figura-2-Diagrama-de-Atividades-modelado-pelo-Grupo-2-antes-da-utilizacao-do-sistema.png

Comportamental

Diagramas de sequência: tipo de diagrama de interação que enfatiza o tipo e a ordem das mensagens passadas entre os elementos durante a execução. Os diagramas de sequência são o tipo mais comum de diagramas de interação, além de ser muito intuitivos para os novos usuários de UML.



Fonte: https://medium.com/documenta%C3%A7ao-uml/introdu%C3%A7%C3%A3o-ao-diagrama-de-sequ%C3%AAncia-1ea5e9563594

Comportamental

Diagramas de caso de uso: documentam os requisitos funcionais para um sistema. Eles fornecem uma visão independente de implementação e do papel que o sistema deve fazer, permitindo ao modelador manter o foco nas necessidades do usuário, ao invés de tê-los nos detalhes de realização.

Por que tantos Diagramas?

O objetivo disso é fornecer múltiplas visões do sistema a ser modelado, analisando-o e modelando-o sob diversos aspectos, procurando-se assim, atingir a completitude da modelagem, permitindo que cada diagrama complemente os outros.

É como se o sistema fosse modelado em camadas, sendo que alguns diagramas enfocam o sistema de forma mais geral, apresentando uma visão externa do sistema, como é o objetivo do Diagrama de Casos de Uso, enquanto quanto outros oferecem um visão de uma camada mais profunda do software.

A utilização de diversos diagramas permite que falhas sejam descobertas, diminuindo a possibilidade da ocorrência de erros futuros.