Engenharia de Software I

Prof. Levi Rodrigues Munhoz

Modelos de processo

Paradigmas:

- Estruturado
- Orientado a Objetos

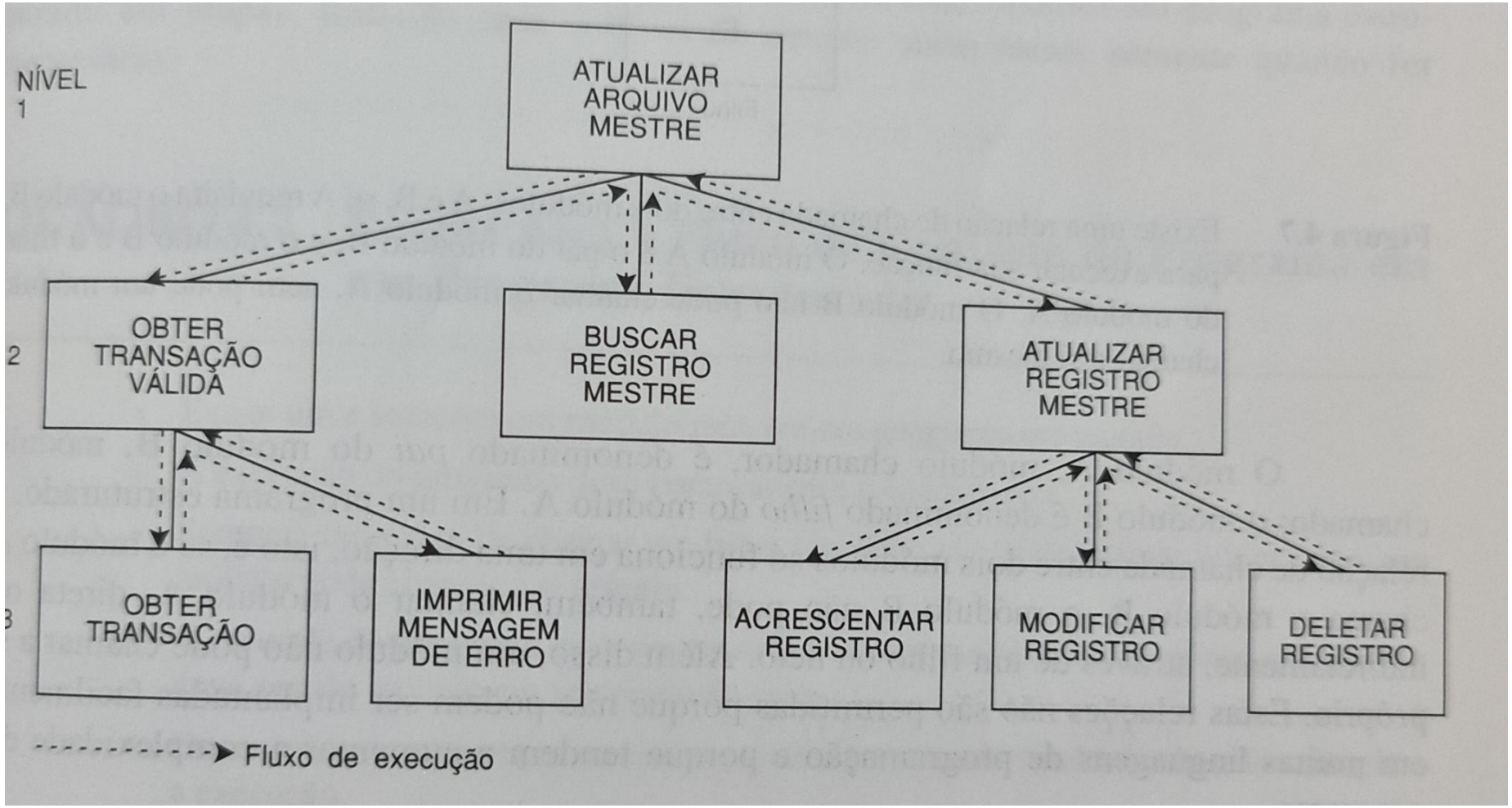
Modelos de processo

Técnicas Estruturadas

- Objetivos
  - Construir programas de alta qualidade que tenham comportamento previsível
  - Construir programas que sejam facilmente modificáveis (manutenção)
  - Simplificar os programas e o seu processo de desenvolvimento

Modelos de processo

- Diagrama de estrutura



Modelos de processo

- Objetivos
  - Conseguir maior previsibilidade e controle no processo de desenvolvimento
  - Acelerar o desenvolvimento de sistemas
  - Diminuir o custo do desenvolvimento de sistemas

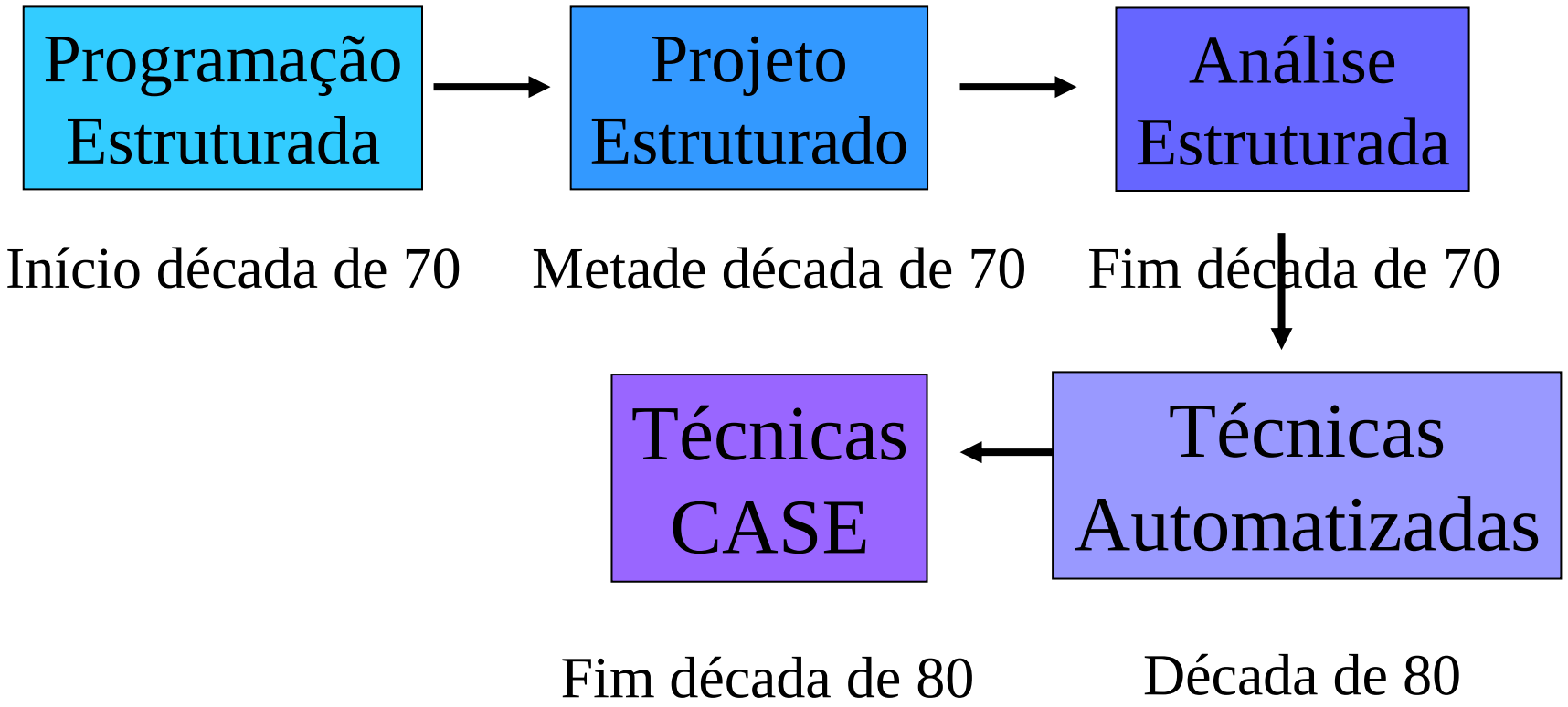
Modelos de processo

- Evolução
  - As técnicas estruturadas evoluíram da Programação Estruturada para técnicas que incluem metodologias de análise, projeto e teste.

Início: fim da década de 60

Modelos de processo

- Evolução



Metodologias e Técnicas Estruturadas

- Evolução
  - Programação Estruturada
    - Focar o programa em si
    - Preocupação com o código do programa
    - Preocupação com a complexidade do programa

Modelos de processo

- Evolução
  - Projeto Estruturado
    - Introduzir organização e disciplina ao projeto
    - Visão de alto nível
    - Conceito de modularização refinado
    - Métricas para qualidade de software

Modelos de processo

- Evolução
  - Análise Estruturada
    - Atenção voltada para a especificação do problema
    - Necessidades dos sistemas
    - Uso de Diagramas de Fluxo de Dados
    - Variedade de técnicas estruturadas

Modelos de processo

Modelagem com técnicas estruturadas

DFD, DER, DTE

Modelos de processo

- Evolução
  - Técnicas Automatizadas
    - Surgiu a necessidade de automação no desenvolvimento de sistemas
    - Uso apropriado do computador, durante as fases de análise e projeto
    - Técnicas de verificação, sendo algumas fundamentadas pela matemática
    - Automatização de código

Modelos de processo

- Evolução
  - CASE
    - Engenharia da Informação (modelos de dados estruturados e modelos da empresa e seus processos)
    - Apoio a projeto com checagem de verificação
    - Especificações a partir das quais o código é gerado automaticamente

Modelos de processo

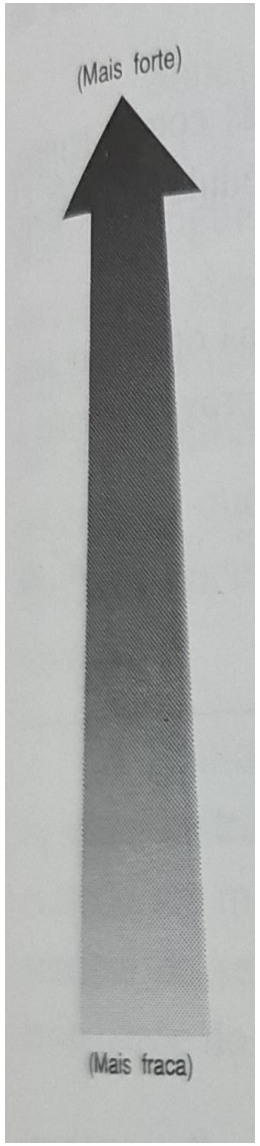
- Princípios
  - Princípios da Técnica Estruturada
- **Princípio da Abstração** - para resolver um problema, separar os aspectos que estão ligados a uma realidade particular, visando representá-lo em forma simplificada e geral
- **Princípio da Formalidade** - seguir uma abordagem rigorosa e metódica para resolver um problema

Modelos de processo

- **Princípio de Dividir-para-Conquistar** - resolver um problema difícil dividindo-o em um conjunto de problemas menores e independentes que são mais fáceis de serem compreendidos e resolvidos

- **Princípio de Organização Hierárquica** - organizar componentes de uma solução em uma estrutura hierárquica tipo árvore. Assim a estrutura pode ser compreendida e construída por nível, cada novo nível acrescentando mais detalhes

Modelos de processo



FUNCIONAL: Cada elemento de um módulo é uma parte necessária e essencial de uma e somente uma função.

SEQUENCIAL: Os elementos de um módulo relacionam-se uns com os outros por executarem diferentes partes de uma sequência de operações onde a saída de uma operação é a entrada da seguinte.

COMUNICACIONAL: Os elementos de um módulo operam todos sobre os mesmos dados.

PROCEDURAL: Os elementos de um módulo são todos parte de um procedimento uma certa sequência de passos que têm de ser executados em uma certa ordem.

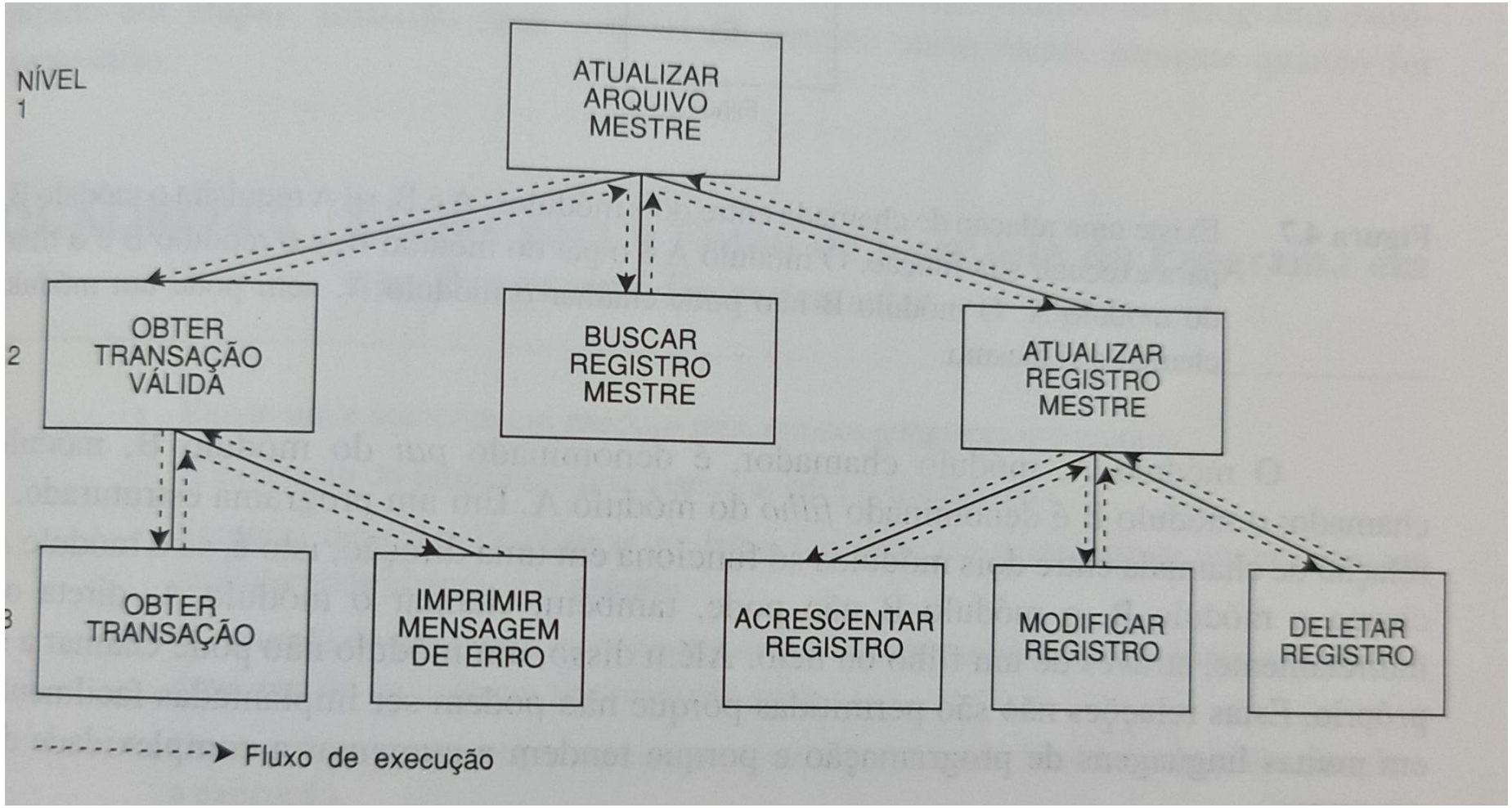
TEMPORAL: Os elementos de um módulo estão relacionados pelo tempo, mas não precisam ocorrer em uma certa ordem ou operar sobre os mesmos dados.

LÓGICA: Os elementos de um módulo são orientados para executar uma certa classe de operações.

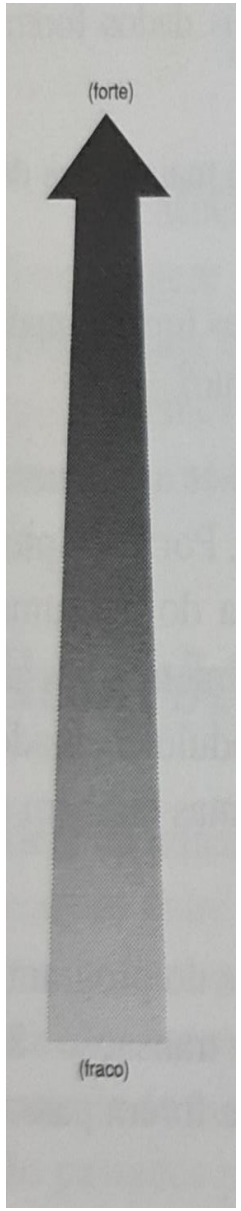
COINCIDENTAL: Os elementos de um módulo não são relacionados por qualquer função, procedimento, dados ou por qualquer outro fator

Modelos de processo

- Diagrama de estrutura



Modelos de processo



ACOPLAMENTO DE CONTEÚDO: Dois módulos possuem acoplamento de conteúdo se um módulo utiliza ou modifica elementos internos de um outro módulo (por exemplo, um desvio, acidental ou não, do código de um módulo para um outro módulo).

ACOPLAMENTO COMUM: Dois módulos possuem acoplamento comum se compartilham as mesmas áreas globais (por exemplo, COMMON do FORTRAN, atributo EXTERNAL do PL/I).

ACOPLAMENTO DE CONTROLE: Dois módulos possuem acoplamento de controle se os dados de um são usados para definir a ordem de execução das instruções do outro (por exemplo, um sinalizador definido em um módulo e testado em uma comparação de um outro módulo).

ACOPLAMENTO DE IMAGEM: Dois módulos possuem acoplamento de imagem se eles se comunicam através de um item de dados composto (por exemplo, registro ou o item de grupo do COBOL). O item de dados composto pode conter partes de dados que não são usadas por um módulo mesmo que lhe sejam passados.

ACOPLAMENTO DE DADOS: Dois módulos possuem acoplamento de dados se eles se comunicam através de uma variável ou de um arranjo (tabela) passados diretamente como um parâmetro entre os dois módulos. O dado é usado em um processamento relacionado com o problema e não com objetivos de controle do programa

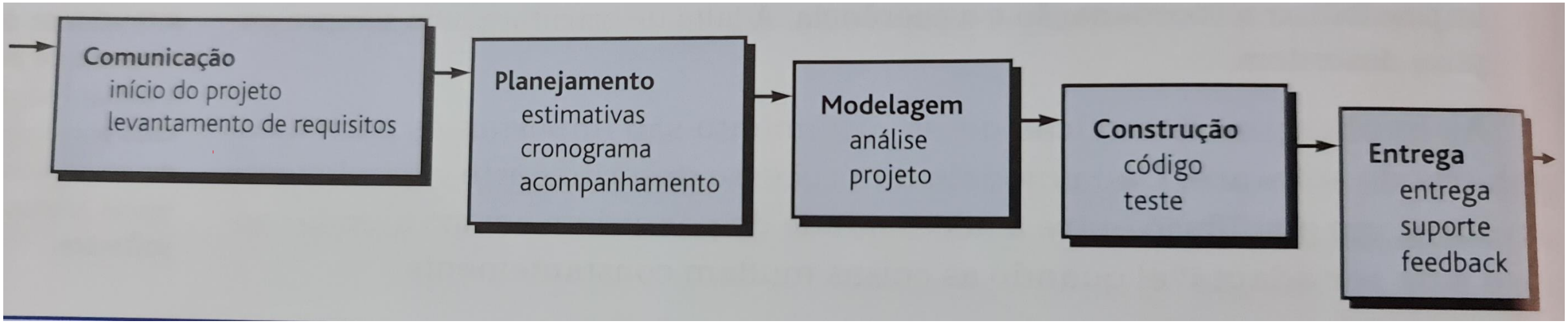
Modelos de processo

1) Modelo de processo prescritivo (tradicionais)

Concentra-se em estruturar e ordenar o desenvolvimento de software. As atividades e tarefas ocorrem sequencialmente, com diretrizes de progresso definidas. Cada modelo de processo também prescreve um fluxo de processo, ou seja a forma pela qual os elementos do processo estão relacionados.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)



Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Comunicação:

- Examinar a solicitação inicial do usuário quanto ao desenvolvimento da aplicação. Assegurar que as informações fornecidas junto com a solicitação estão completas e precisas. Caso contrário, contatar o solicitante para esclarecer ambiguidades.
- Com base na breve descrição da solicitação de serviço para um sistema, identificar todos os departamentos e sistemas relacionados que provavelmente serão afetados pela solicitação do projeto proposto.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Comunicação:

- Para cada provável departamento de usuário envolvido, identificar os nomes dos representantes de usuários que devem ser contatados durante o projeto. Confirmar suas participações e solicitar concordância quanto a papéis, responsabilidades e disponibilidades previstas.
- Conduzir diversas entrevistas e reuniões com indivíduos que representem cada departamento de usuário afetado.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Comunicação:

- Descrever o problema que está sendo experimentado no momento.
- Descrever as novas oportunidades que estão exigindo um novo sistema.
- Documentar todas as informações obtidas durante as entrevistas e reuniões com usuários.
- Verificar os fatos obtidos e confirmar suas descobertas com os entrevistados e seus respectivos superiores.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Comunicação:

- Se alguns problemas e oportunidades parecerem conflitantes, documentar os assuntos relacionados e submetê-los às chefias dos usuários para explicações e solução.
- Atribuir uma prioridade a cada problema /oportunidade que for identificado (a) durante os processos preliminares de obtenção de fatos.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Comunicação:

- Rever a lista de problemas/oportunidades declarados (as) e resumir a situação atual. Derivar uma descrição consolidada que realce todas as necessidades dos usuários.
- Rever as necessidades dos usuários quanto aos problemas/oportunidades identificados, para garantir que realmente reflitam a situação atual e que todos os grupos afetados na empresa foram contatados.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento:

- Definir a missão do sistema proposto.
- Documentar os objetivos de negócio que devem ser alcançados com a instalação do sistema.
- Priorizar esses objetivos por ordem de importância e grau de necessidade.
- Para cada objetivo listado, identificar o departamento de negócio específico associado a ele.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento:

- Ensaiar tentativas de missão do sistema e objetivos de negócio do projeto para garantir que eles estejam completos e sejam expressos da melhor forma possível em termos quantificáveis.
- Criar um diagrama de contexto
- Criar um diagrama de contexto de dados
- Identificar restrições e riscos do projeto

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento:

- Identificar as funções de negócio e as entidades de dados de alto nível que se encaixem nos limites do sistema proposto. Assegurar que elas foram incluídas no escopo deste projeto.
- Identificar as interligações do sistema que, apesar de poderem ser afetadas pelo sistema proposto, não podem ser modificadas por este projeto. Documentar a lógica associada às limitações incorridas e descrever como possivelmente isso pode afetar o projeto.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento:

- Revisar todos os produtos técnicos desenvolvidos até o momento e modificá-los à luz dos limites do projeto proposto e das restrições e riscos declarados do projeto, sempre que aplicável.
- Confirmar os limites do projeto para garantir que ele realmente reflete as principais funções de negócio a serem fornecidas pelo sistema, bem como as entidades de dados que estão sob sua jurisdição.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento:

- Identificar e descrever rapidamente as soluções alternativas de implementação do sistema que possam fornecer um método de solução para os problemas existentes ou atingir as oportunidades existentes (ao mesmo tempo que atende aos objetivos de negócio do projeto).

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento:

- Estabelecer prioridades para as soluções de implementação do sistema proposto com uma breve justificativa para a sua classificação. A alternativa preferida é identificada como a solução "recomendada" para atingir os objetivos de negócio e satisfazer as necessidades dos usuários

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento:

- Se as soluções propostas forem de longo prazo, identifique quaisquer soluções em potencial de curto prazo que possam reduzir os problemas nesse interim.
- Investigue a conveniência e a viabilidade de se montar um protótipo do sistema. Se aplicável, recomende isso como alternativa ao tradicional ciclo de vida de desenvolvimento.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento:

- Confirme as alternativas propostas para garantir que cada uma delas atenda aos objetivos de negócio estabelecidos

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento: Estratégia preliminar de custo-benefício

Estimativas de custos.

- Identificar a ordem de magnitude dos custos associados à operação do sistema existente em uma base anual.
- Identificar, se possível neste estágio, a ordem de magnitude dos custos operacionais de cada alternativa de sistema proposta numa base anual.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento: Estratégia preliminar de custo-benefício

Estimativas de custos.

- Identificar a ordem estimada de magnitude dos custos de desenvolvimento de cada solução de sistema proposta.
- Descrever as técnicas que serão usadas para gerar as estimativas de custo.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento: Estratégia preliminar de custo-benefício

Benefícios estimados.

- Identificar a ordem de magnitude das estimativas de benefícios associados de cada solução proposta. Classifique-as em dois grupos: benefícios tangíveis e benefícios intangíveis.
- Para cada benefício tangível, quantifique os valores monetários das reduções esperadas dos custos operacionais ou dos aumentos de receita. Identifique o período de retorno projetado associado ao novo sistema.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento: Estratégia preliminar de custo-benefício

Benefícios estimados.

- Para cada benefício intangível, descreva os valores adicionados associados à implementação do sistema e forneça as justificativas apropriadas onde for necessário.
- Descreva as técnicas usadas para quantificar os benefícios.
- Confira o documento de análise preliminar de custo-benefício.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento: Estratégia preliminar de custo-benefício

Análise de custo-benefício. Abaixo, mostramos uma lista incompleta dos vários tipos de custos e benefícios que podem ser considerados no projeto.

Custos operacionais (sistemas atuais e propostos). Os custos operacionais podem ser agrupados nas seguintes categorias:

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento: Estratégia preliminar de custo-benefício

CUSTOS DE INFORMÁTICA

- Custos associados ao pacote (isto é, licença de software).
- Custos da equipe de computação dedicada.
- Custos de armazenamento de dados.
- Custos de comunicação de dados e distribuição.
- Custos de manutenção.
- Custos de utilização do computador principal.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento: Estratégia preliminar de custo-benefício

CUSTOS DE PESSOAL

- Pessoal de operação dos computadores
- Pessoal do usuário.
- Custos de treinamento.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento: Estratégia preliminar de custo-benefício

CUSTOS DE SUPRIMENTOS

- Formulários.
- Papéis.

CUSTOS DIVERSOS

- Aluguel de espaço
- Mobiliário de escritório.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento: Estratégia preliminar de custo-benefício

Benefícios. Os benefícios são classificados em duas categorias.

Tangíveis (isto é, quantitativos) e intangíveis (isto é, qualitativos)

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento: Estratégia preliminar de custo-benefício

BENEFÍCIOS TANGÍVEIS

Os benefícios quantitativos são os que podem ter um valor monetário diretamente associado. Incluem, entre outros:

- Novas fontes de renda.
- Aumento de lucros.
- Redução de custos (equipe).

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento: Estratégia preliminar de custo-benefício

BENEFÍCIOS TANGÍVEIS

- Redução dos custos de manutenção.
- Componentes do novo sistema que já sejam comercializados na indústria.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento: Estratégia preliminar de custo-benefício

BENEFÍCIOS INTANGÍVEIS

Os benefícios qualitativos são aqueles para os quais nem sempre é possível associar um valor monetário direto e tangível. Entre outras coisas, incluem:

- Serviço melhorado.
- Informações mais precisas e a tempo.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Planejamento: Estratégia preliminar de custo-benefício

BENEFÍCIOS INTANGÍVEIS

- Controles melhorados.
- Maior flexibilidade.
- Melhor margem de competitividade.
- Melhoria do moral dos empregados.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Analisar o Sistema atual

- Revisar os produtos técnicos produzidos durante a fase de pesquisa.
- Revisar toda a documentação colhida para atualizar o sistema atual.
- Se necessário, completar o processo de obtenção de informações coletando toda a documentação pertinente que descreva as operações de talhadas do sistema atual.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Analisar o Sistema atual

- Se necessário, criar um inventário da documentação detalhada obtida do sistema atual.
- Analisar as informações do sistema atual e consolidar os problemas, oportunidades e necessidades dos usuários que foram documentados durante a fase de pesquisa.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Analisar o Sistema atual

- Discutir com os usuários os principais pontos fortes e fracos do sistema atual. Se apropriado, identificar as características essenciais do sistema atual que os usuários gostariam de manter no novo sistema.
- Conferir a documentação do sistema atual para garantir que reflita adequadamente o ambiente existente e validar os problemas, oportunidades e necessidades consolidadas dos usuários.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Analisar o Sistema atual

Obtenção da documentação existente. A primeira tarefa da fase de análise preliminar é avaliar o quanto o sistema atual serve aos usuários em seus ambientes de negócio atuais.

Ambiente manual dos usuários

- Diagramas existentes que retratem a estrutura hierárquica da organização dos usuários.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Analisar o Sistema atual

- Conjunto de documentos de entrada/saída que é usado para executar as atividades manuais do sistema. Pode incluir diferentes tipos de formulários, relatórios ou procedimentos departamentais.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Analisar o Sistema atual

- Descrição dos atuais arquivos manuais.
- Documentos que descrevam a natureza das interfaces manuais existentes entre o ambiente dos usuários e ambientes de outros grupos dentro ou fora da organização.
- Documentos que descrevam as políticas corporativas que regulam as atividades dos negócios exercidas nas áreas dos usuários.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Analisar o Sistema atual

Ambientes automatizados

- Principais características de configurações de hardware/software/rede em que o sistema rode.
- Programas batch do sistema, junto com amostras de relatórios produzidos, bem como identificação dos receptores.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Analisar o Sistema atual

- Transações on-line do sistema, junto com layouts de suas telas relacionadas.
- Operações do sistema em tempo real.
- Estruturas de arquivos e bancos de dados do sistema, junto com amostras de seus registros ou layouts de segmentos.
- Descrição das interfaces do sistema.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Analisar o Sistema atual

- Documentos-fontes usados para alimentar o sistema computadorizado, junto com amostras de cada documento.
- Uma cópia dos manuais de sistema do usuário.
- Documentação do sistema que descreva os principais componentes do sistema atual.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Modelo conceitual de processos

- Revisar o modelo conceitual de processos antigos do sistema, se existirem.
- Com base nas informações obtidas até o momento sobre o sistema existente e seu ambiente, revise o diagrama de contexto de sistema montado durante a fase de planejamento quanto à sua precisão e inteireza.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Modelo conceitual de processos

- Divida o sistema proposto em componentes subordinados, nivelando cada diagrama lógico de fluxo de dados, do nível mais alto para o mais baixo, conforme a seguir: diagrama de sistema, diagramas de subsistema, diagramas de funções.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Modelo conceitual de processos

- Mapeie o modelo conceitual de processo da aplicação que está sendo desenvolvida para se apropriar dos componentes de modelo corporativo de processos. Resolva quaisquer inconsistências que existam entre a aplicação relacionada e os modelos corporativos de processos.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Modelo conceitual de processos

- Forneça uma descrição textual de todos os elementos gráficos mostrados no modelo conceitual de processos (isto é, os diagramas de contexto, de sistema, de subsistema e de fluxo de dados de funções).
- Teste o modelo conceitual de processos para garantir que represente completa e precisamente o conjunto de funções essenciais que devem ser executadas pelo sistema proposto e que cubra de forma adequada todas as necessidades de funções do usuário.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Modelo conceitual de dados

- Revisar o antigo modelo conceitual de dados, se existir.
- Finalizar a construção do modelo conceitual de dados, usando a técnica de modelagem entidade-relacionamento.
- Resolver as anomalias que possam ser detectadas durante o desenvolvimento do modelo conceitual de dados.
- Documentar cada item gráfico mostrado no diagrama entidade-relacionamento com uma descrição textual correspondente.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Modelo conceitual de dados

- Mapear o modelo conceitual de dados da aplicação em desenvolvimento para os componentes apropriados do modelo corporativo de dados. Verificar se as entidades, os elementos de dados e relacionamentos são consistentes com os apresentados no modelo corporativo de dados. Resolver quaisquer inconsistências que surjam entre os modelos de dados relacionados com a aplicação e com a corporação.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Modelo conceitual de dados

- Testar o modelo conceitual de dados e seus componentes para garantir que reflita precisamente os dados usados pelo sistema e que esteja completo.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Modelo conceitual de dados e processos

- Verificar se não existem inconsistências entre o modelo conceitual de processos (junto com suas características de tempo real, quando apropriado) e o modelo conceitual de dados.
- Verificar se o modelo conceitual de dados pode satisfazer todos os requisitos identificados no fluxos e depósitos de dados descritos no modelo conceitual de processos.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Modelo conceitual de dados e processos

- Avaliar o nível de estabilidade dos modelos conceituais de processos e dados no contexto do ambiente do usuário, no presente e no futuro.
- Se necessário, teste os modelos conceituais consolidados de dados e processos

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Modelo conceitual de dados e processos

- Isso é feito principalmente com base na probabilidade de introduzir novos requisitos ou funções de dados no ambiente em um tempo razoável. Se necessário, ajuste os modelos para suportar futuros requisitos de negócio e de informações que sejam previstos pelos usuários e documente as razões para essa abordagem.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Identificar os limites preliminares do sistema

Para cada cenário de limites, executar as seguintes atividades.

- ✓ Identificar no modelo conceitual de processos as unidades de funções/trabalho que são candidatas apropriadas à informatização.
- ✓ Identificar no modelo conceitual de dados as entidades de dados que são candidatas apropriadas à informatização.
- ✓ Avaliar as vantagens e desvantagens de cada cenário de automatização proposto, julgando sua capacidade de satisfazer os requisitos de usuários. Documentar as descobertas.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Identificar os limites preliminares do sistema

- Recomendar o cenário preferido de automação, descrevendo as razões para sua escolha em relação às outras alternativas.
- Executar as seguintes subtarefas para as partes automatizadas do sistema:

✓ Identificar as unidades de funções/trabalho que provavelmente serão executadas nos modos de processamento batch, on-line ou em tempo real.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Identificar os limites preliminares do sistema

- ✓ - Identificar as entidades de dados do modelo conceitual de dados que provavelmente serão implementadas com ou sem o uso de tecnologia de bancos de dados.
- Descrever as características operacionais gerais que são recomendadas para implementar as diversas partes automatizadas do sistema, como as seguintes:

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Identificar os limites preliminares do sistema

- ✓ Processamento centralizado, descentralizado ou distribuído de funções e dados.
- ✓ Implementação de algumas partes do sistema automatizado ou todo ele em micros, minis ou mainframes.

- Testar os limites automatizados/manuais preliminares do sistema.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Requisitos preliminares de desempenho

- Identificar os requisitos de nível mais alto que serão usados para medir o desempenho do sistema em relação ao seguinte:

✓ Tempo médio de resposta para as partes on-line e em tempo real do sistema.

✓ Tempo médio de execução e de rotatividade de tarefas que se espera do processamento batch do sistema.

✓ Disponibilidade global do sistema.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Requisitos preliminares de controle do sistema

- Segurança
- Integridade
- Auditabilidade
- Principais situações de recuperação de desastres

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Requisitos preliminares de armazenamento de dados

- Identificar os requisitos preliminares de armazenamento de dados para o sistema. Incluir o tamanho inicial dos bancos de dados e/ou arquivos que serão criados ou atualizados pelo sistema. Incluir também as estimativas de crescimento da capacidade de armazenamento, cobrindo um período de pelo menos dois a cinco anos.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Requisitos preliminares de armazenamento de dados

- Identificar os requisitos esperados de retenção de dados do sistema, junto com as técnicas usadas para derivar estimativas de espaço.
- Testar os requisitos preliminares de armazenamento de dados do sistema.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Requisitos preliminares de hardware e software e redes

- Identificar as novas classes de hardware necessárias para suportar o novo sistema.
- Identificar as novas classes de software necessárias para suportar as funções e dados mecanizadas do novo sistema.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Requisitos preliminares de hardware e software e redes

- Identificar as localizações geográficas onde o novo sistema será usado. Descrever em termos gerais como as funções e dados serão processados e/ou distribuídos para cada localização. Descrever os novos recursos de comunicação de dados necessários para suportar as operações do sistema proposto.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Requisitos preliminares de hardware e software e redes

- Consolidar os requisitos de hardware/software para garantir que eles sejam compatíveis entre si e com os recursos de hardware/software da organização.
- Testar os requisitos preliminares de hardware/software.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Criar um modelo funcional de Processos

- Revisar os produtos técnicos construídos de forma preliminar e avançar para o detalhamento, criando diagramas de fluxos de dados de cada unidade de trabalho. Além da parte gráfica, documentar em detalhes cada função.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Criar um modelo funcional de Dados

- Revisar os produtos técnicos construídos de forma preliminar e avançar para o detalhamento e normalização, criando diagramas de entidade e relacionamento e de cada unidade de trabalho. Alimentar o dicionário de dados.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Desenvolver uma estratégia de conversão de dados preliminar

- Identificar nas classes de dados atuais, o que pode ser recuperado e adaptado para o novo sistema, evitando redigitação. Definir um cronograma de atividades e responsabilidades.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Desenvolver uma estratégia de testes preliminar

- Identificar as unidades a serem testadas e que tipo de testes serão feitos. Definir como as massas de teste serão construídas considerando as necessidades de teste de cada unidade.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Desenvolver uma estratégia de treinamento preliminar

- Identificar as principais mudanças, categorias de pessoal e formas de treinamento a serem aplicadas. Definir um cronograma de treinamento.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Refinar os requisitos de Hardware/Software e Redes

- Com base nos volumes de dados, processos e pessoas, definir detalhes de necessidades em termos de hardware, software básico, número de licenças e redes.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Refinar os requisitos de controle do sistema

- Segurança → Níveis de usuários e acessos a dados e funções
- Integridade → Dos bancos de dados e evitar ao máximo digitação
- Auditabilidade → Trilhas de auditoria e detecção de fraudes
- Principais situações de recuperação de desastres → Criar os manuais do plano de contingência e possíveis parceiros.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Refinar os requisitos de desempenho

- Baseado nos volumes de dados, necessidades de impressão e transmissão via redes, estabelecer tempo de resposta para cada requisito funcional.
- Identificar requisitos críticos deste tema.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Definir padrões de projeto e da arquitetura do sistema

- Criar padrões de telas, relatórios, formulários, diálogos homem-máquina, formatos de mensagens de erro. Padrões de reuso, regras de segurança de acesso a dados e a banco de dados.
- Adequar aos padrões vigentes na organização e possível revisão desses padrões em função das novas necessidades.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Criar modelo físico de dados

- Com base nos diagramas de entidade e relacionamento, definir layout de cada tabela, bem como as chaves de acesso. Tipos de dados, tamanho, máscara de edição de cada atributo.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Criar modelo físico de processos

- Derivar um primeiro diagrama de estrutura de cada programa, considerar grau de coesão e acoplamento
  - Detalhar a lógica das funções mais complexas e o acesso a dados.
  - Incluir módulos para validar dados de entrada
  - Desenhar layout de telas e relatórios;
  - Definir com os dados serão acessados em cada programa

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Desenvolver a estratégia de conversão de dados detalhada

- Para cada tabela do novo sistema, identificar:
  - A origem dos dados a serem recuperados e localização.
  - Definir programas novos, específicos para a conversão;
  - Identificar aquilo que deve ser digitado, pois não há como recuperar;
  - Filtros e adaptações em dados;
  - Identificar responsáveis e estabelecer um cronograma atualizado.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Definir estratégia detalhada de treinamento

- Definir cursos que serão ministrados, plano de ensino, cronograma e aplicação de cada curso, necessidades de hardware, software, matérias. Quem dará o treinamento e quais usuários serão treinados.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Modelagem: Análise e Projeto

Definir estratégia detalhada de testes

- Para cada programa, definir:
  - Massa de teste, sequência de teste, formato do teste de unidade, teste de integração, teste de aceitação, estres e regressão

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Construção: Instalar os equipamentos e recursos de hardware software e redes

- Garantir que cada local em que o sistema será instalado tenha sido adequadamente preparado para aceitar os equipamentos e recursos de hardware/software/rede que serão usados para desenvolver e testar o sistema.
- Receber os equipamentos e recursos de hardware/software/rede necessários.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Construção: Criar o ambiente de testes do sistema

- Instalar os equipamentos de hardware.
- Instalar os recursos de software.
- Verificar os equipamentos e recursos de hardware/software/rede necessários executando casos de teste de instalação.
- Notificar a equipe de desenvolvimento sobre o sucesso da instalação dos equipamentos e recursos de hardware/software/rede.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Construção: Criar o ambiente de testes do sistema

- Analisar a aplicação dos requisitos de teste definidos na estratégia detalhada de teste dos produtos do sistema.
- Criar diferentes conjuntos de bibliotecas necessários para suportar adequadamente cada nível de teste em particular (isto é, unidade, integração de sistema, aceitação pelos usuários, aceitação em produção).

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Construção: Criar o ambiente de testes do sistema

- Se necessário, desenvolver os procedimentos manuais/automatizados de controle de bibliotecas necessários para a administração apropriada de cada ambiente de teste específico.
- Codificar e testar todas as definições de bancos de dados, blocos de controle de programas, definições de arquivos e utilitários de cópia de segurança/recuperação que serão usadas no ambiente de teste.
- Se necessário, inicializar os arquivos e bancos de dados de teste.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Construção: Criar o ambiente de testes do sistema

- Onde aplicável, preencher os arquivos/bancos de dados de teste da aplicação dos ambientes de teste com os dados de teste ou valores padrões apropriados.
- Se necessário, criar um guia para o ambiente de teste, descrevendo como usar efetivamente os diversos recursos fornecidos no ambiente de teste.
- Testar o ambiente de teste do sistema.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Construção: Codificar programas da aplicação

- Analisar os fluxos e especificações originais da estrutura do programa.
- Se ainda não tiver sido feito, codificar os layouts de tela e relatórios manipulados pelo programa.
- Desenvolver o código-fonte do programa na linguagem escolhida para desenvolver a aplicação.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Construção: Codificar programas da aplicação

- Obter uma compilação sem erros dos módulos-fontes dos programas.
- Documentar cada programa de acordo com os padrões prescritos pela organização.
- Testar cada código-fonte de programa e sua documentação.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Construção: Executar os testes de unidade

- Montar casos de teste representativos da lógica interna de cada programa.
- Desenvolver as instruções de controle de execução de tarefas necessárias para testar o programa.
- Executar os testes das unidades nos módulos dos programas.
- Analisar os resultados dos testes e compará-los com os resultados esperados.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Construção: Executar os testes de unidade

- Se forem encontrados erros, determinar suas causas. Se for um erro de construção de unidade (por exemplo, erros de codificação), corrigi-lo e reexecutar os testes de unidade do programa. Se os erros forem decorrentes de análise incorreta ou de especificações de projeto, seguir os procedimentos de controle de mudança prescritos para o projeto.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Construção: Executar os testes de unidade

- Se não forem encontrados erros, transferir os programas testados para as bibliotecas de teste de integração do sistema.
- Documentar os resultados dos testes de unidades.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Construção: Executar os testes de integração do sistema

- Completar a montagem dos casos de teste de integração do sistema.
- Desenvolver as instruções de controle de tarefas detalhadas necessárias para testar as rotinas de tarefas batch.
- Executar os testes de integração do sistema.
- Analisar as saídas dos testes e compará-las com os resultados esperados.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Construção: Executar os testes de integração do sistema

- Se forem encontrados erros, determinar suas causas. Se necessário, devolva o programa com defeito para o ambiente de teste e emita uma solicitação de alteração de sistema para resolver os problemas encontrados durante o processo de teste de integração do sistema.
- Se não forem encontrados erros, transfira o(s) programa(s) da nova versão para as bibliotecas de teste de aceitação pelo usuário.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Construção: Executar os testes de integração do sistema

- Documente os resultados do teste de integração do sistema.
- Procedimentos de cópias de segurança e recuperação de arquivos/bancos de dados.
- Recursos de auditoria, controle e segurança aplicáveis a todo o sistema.
- Características desejadas de processamento, capacidade e desempenho do sistema.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Construção: Executar os testes de integração do sistema

- Interfaces externas com outros sistemas.
- Rotinas de tarefas batch e suas dependências.
- Necessidades temporárias e permanentes de armazenamento do sistema.
- Utilidade, validade, correção e facilidade de entendimento dos relatórios produzidos pelo sistema e pelos diversos diálogos homem-máquina.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega: Preparar um esboço do pacote de manuais do sistema

- Revisar os índices preliminares desenvolvidos durante a fase de projeto para cada tipo de manual do sistema. Verificar as suas organizações e se são completos com base nas informações obtidas até o momento sobre o novo sistema.
- Esboçar o conteúdo de cada manual individual do sistema.
- Testar os quatro esboços de manuais quanto à clareza e conteúdo.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Preparar um esboço do pacote de treinamento

- Revisar a estratégia detalhada de treinamento do sistema com base em todas as informações produzidas até o momento sobre o novo sistema.
- Desenvolver esboços do material necessário para treinar a equipe inicial de teste dos usuários.
- Se aplicável, instalar e testar as ferramentas de treinamento do software.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Entrega : Preparar um esboço do pacote de treinamento

- Testar os esboços do pacote de treinamento para verificar se são precisos e se atendem aos objetivos declarados de treinamento.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Treinar a equipe usuária de teste inicial

- Treinar a equipe usuária inicial que executará os testes de aceitação.
- Produzir o relatório final do treinamento inicial.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Executar os testes de aceitação pelos usuários

- Completar a montagem de casos de teste de aceitação por representantes dos usuários.
- Executar os testes de aceitação pelos usuários de acordo com a estratégia detalhada de testes do sistema.
- Analisar as saídas dos testes e compará-las com os resultados esperados.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Executar os testes de aceitação pelos usuários

- Se forem encontrados erros, determinar suas causas. Se exigido, devolver os programas com defeitos para o ambiente de testes de unidades. Se necessário, emitir uma solicitação de alteração de sistema para resolver os problemas encontrados durante o processo de testes de aceitação pelos usuários

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Executar os testes de aceitação pelos usuários

- Se não forem encontrados erros, transferir os programas para as bibliotecas de testes de aceitação pela produção.
- Documentar os resultados dos testes de aceitação pelos usuários.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Finalizar o pacote de manuais do sistema

- Revisar os esboços do pacote de manuais e completar todas as seções apropriadas, onde for aplicável.

- Preparar a lista detalhada de distribuição de todos os departamentos e pessoal relacionado que receberão os manuais individuais do sistema. Identificar a quantidade total de cópias necessárias para cada área. É possível que os manuais estejam on-line e em formato help ou hint (dica) nas telas

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Finalizar o pacote de treinamento

- Revisar e finalizar o conteúdo do pacote de treinamento do sistema com base nas informações obtidas até o momento sobre o sistema.

- Avaliar a duração de cada tipo particular de sessão de treinamento e preparar o planejamento detalhado de treinamento, juntamente com uma lista completa do pessoal a ser treinado.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Finalizar o pacote de treinamento

- Se necessário, confirmar a disponibilidade dos diversos recursos de treinamento (salas de aula, equipamento de treinamento etc.).
- Testar intensivamente o pacote final de treinamento com os representantes de usuários e de operação/manutenção de sistemas.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Entrega : Treinar formalmente todos os usuários/pessoal de sistemas

- Finalizar o plano detalhado de treinamento de todo o pessoal afetado.
- Treinar a equipe usuária que operará diretamente o sistema e seus diversos níveis de administração, se necessário.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Entrega : Treinar formalmente todos os usuários/pessoal de sistemas

- Treinar a equipe de operação do sistema e suas gerências.
- Se necessário, treinar o pessoal e as gerências de manutenção de sistemas.
- Produzir o relatório completo do final do treinamento.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Executar um teste de aceitação em produção

- Completar a montagem de casos de testes de aceitação em produção.
- Executar os testes de aceitação em produção de acordo com a estratégia detalhada de testes do sistema.
- Analisar as saídas dos testes e compará-las com os resultados esperados.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Executar um teste de aceitação em produção

- Se forem detectados erros, determinar suas causas. Se necessário, devolver os programas com defeito para o ambiente de teste de unidades e emitir uma solicitação para resolução dos problemas detectados no processo de aceitação em produção.
- Se não forem detectados erros, transferir o(s) programa(s) para as bibliotecas de produção.
- Documentar os resultados dos testes de aceitação em produção.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Instalar os equipamentos e recursos de hardware/software/rede nos locais dos usuários

- Verificar se cada local de instalação de usuário foi adequadamente preparado para aceitar os novos equipamentos e recursos.
- Receber e instalar os novos equipamentos e recursos de hardware/software/rede nos locais dos usuários.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Instalar os equipamentos e recursos de hardware/software/rede nos locais dos usuários

- Testar os equipamentos e recursos com os casos de teste de instalação para garantir que funcionem apropriadamente nos modos de operação isolado e integrado.
- Se necessário, verificar se os diversos suprimentos de informática foram entregues em quantidade suficiente em cada área de usuário.
- Gerar o relatório completo do final da instalação de hardware/software/rede.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Executar a conversão dos arquivos/bancos de dados de produção existentes

- Se aplicável, verificar se os dados manuais do sistema antigo foram capturados apropriadamente dentro do novo sistema.
- Executar os programas de conversão na sequência proposta no plano e estratégia final de conversão.
- Verificar se os conteúdos dos novos arquivos/bancos de dados de produção estão completos e precisos.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Executar a conversão dos arquivos/bancos de dados de produção existentes

- Se houver discrepâncias, identificar suas causas e aplicar as medidas corretivas apropriadas.
- Se necessário, salvar os arquivos e bancos de dados do sistema antigo para consultas futuras e para fins de auditoria.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Entrega : Migrar o novo sistema para produção

- Transferir todo o sistema do ambiente de desenvolvimento para o ambiente de produção.
- Verificar se todos os elementos do sistema foram transferidos com sucesso para as bibliotecas de produção.
- Distribuir os diversos manuais do sistema para os representantes oficiais dos usuários e de sistemas.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Migrar o novo sistema para produção

- Ativar o sistema em produção e verificar se o seu nível de operabilidade é adequado.
- Entregar o sistema para os usuários.
- Quando aplicável, desativar o sistema antigo.
- Completar o relatório de migração do sistema para a produção.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Executar otimização pós-implementação

- Avaliar o desempenho global e o nível de operabilidade do sistema como um todo, quanto a:
  - Tempo de resposta/execução de programas
  - Tempo de execução de utilitários de software (cópias de segurança/recuperação, banco de dados etc.)
  - Recursos de segurança.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Executar otimização pós-implementação

- Equipamento de computação instalado.
- Instruções de operação de computadores e de controle de execução
- Documentação geral de usuário/sistemas.
- Rede de comunicação do sistema.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Entrega : Executar otimização pós-implementação

- Identificar as áreas específicas em que os requisitos originais do sistema não foram seguidos à risca. Determinar as causas dos problemas e aplicar as correções apropriadas.
- Completar a documentação do sistema, se isso ainda não foi feito.
- Completar o relatório de otimização do sistema em produção.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Entrega: Desenvolver a estratégia de manutenção do sistema

- Desenvolver a estratégia de manutenção do sistema, levando em consideração:
  - Solicitações de mudança que foram aprovadas oficialmente durante o ciclo de desenvolvimento, mas que foram postergadas de propósito para a fase de manutenção.
  - - Recursos totais de pessoal de manutenção da área de sistemas necessários para apoiar os diversos setores de usuários que serão atendidos pelo novo sistema.

Modelos de processo

1.1 Cascata (ciclo de vida clássico)

Entrega: Desenvolver a estratégia de manutenção do sistema

- Desenvolver a estratégia de manutenção do sistema, levando em consideração:
  - Solicitações de mudança que foram aprovadas oficialmente durante o ciclo de desenvolvimento, mas que foram postergadas de propósito para a fase de manutenção.
  - - Recursos totais de pessoal de manutenção da área de sistemas necessários para apoiar os diversos setores de usuários que serão atendidos pelo novo sistema.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

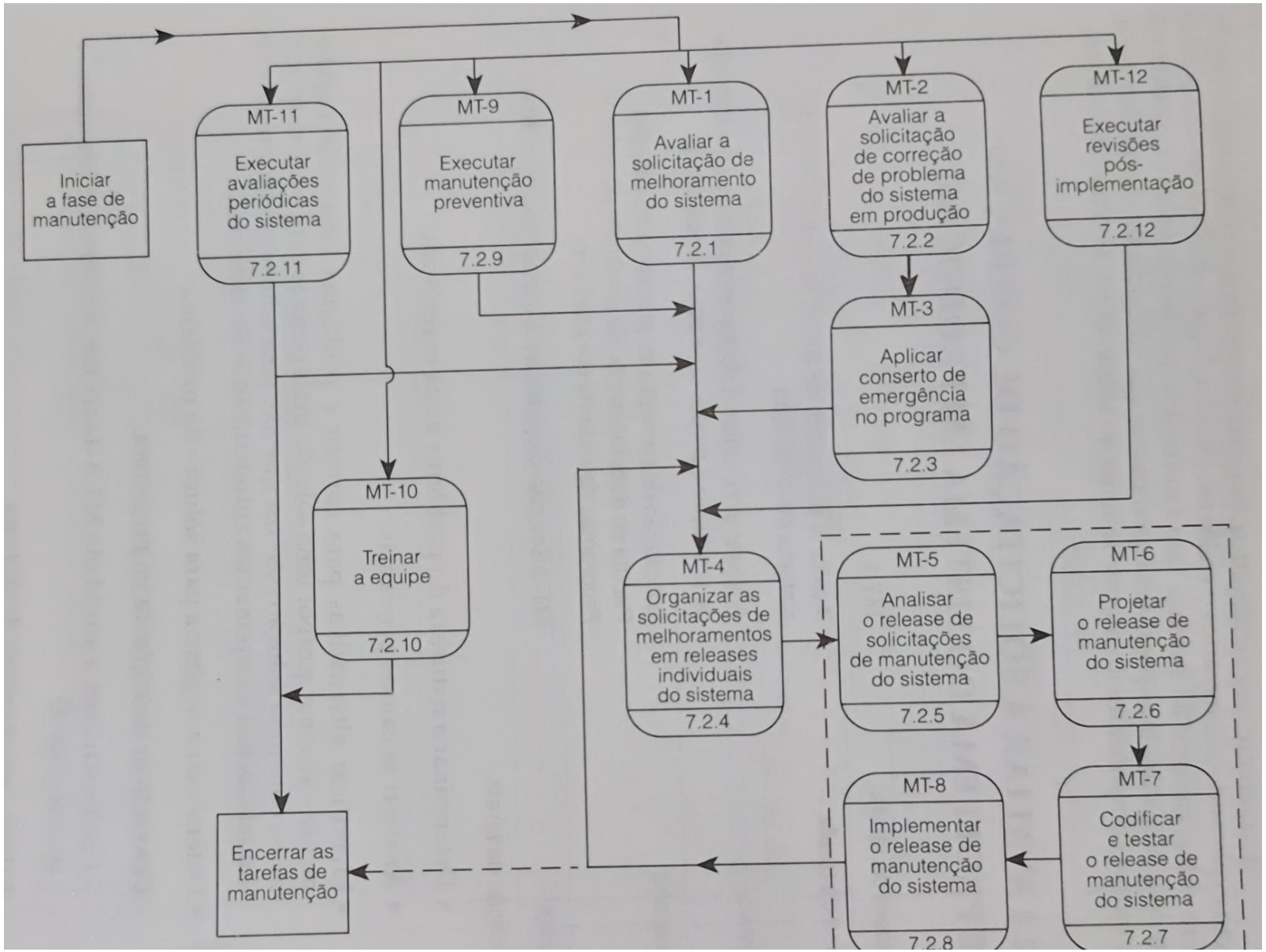
Entrega: Desenvolver a estratégia de manutenção do sistema

- Solicitações de mudança que foram aprovadas oficialmente durante o ciclo de desenvolvimento, mas que foram postergadas de propósito para a fase de manutenção.

- - Recursos totais de pessoal de manutenção da área de sistemas necessários para apoiar os diversos setores de usuários que serão atendidos pelo novo sistema.

- Testar a estratégia de manutenção do sistema.

Modelos de processo



Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Quando o modelo em cascata é vantajoso

- Nos casos em que os requisitos de um problema são bem compreendidos, em algumas situações quando as adaptações ou **aperfeiçoamentos bem-definidos precisam ser feitos em um sistema existente** (por exemplo, uma adaptação de um software contábil, exigida devido à mudanças governamentais). Outra situação seria quando o software a ser desenvolvido tem **requisitos bem definidos** e são razoavelmente **estáveis**.

Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Desvantagens

Sendo o paradigma mais antigo, ao longo dos anos sofreu muitas críticas. Lista-se abaixo os problemas muitas vezes encontrados:

1. Projetos reais raramente seguem o fluxo sequencial proposto pelo modelo. Embora o modelo linear possa conter iterações, ele o faz indiretamente. Como consequência, mudanças podem provocar confusão à medida que a equipe de projeto prossegue.

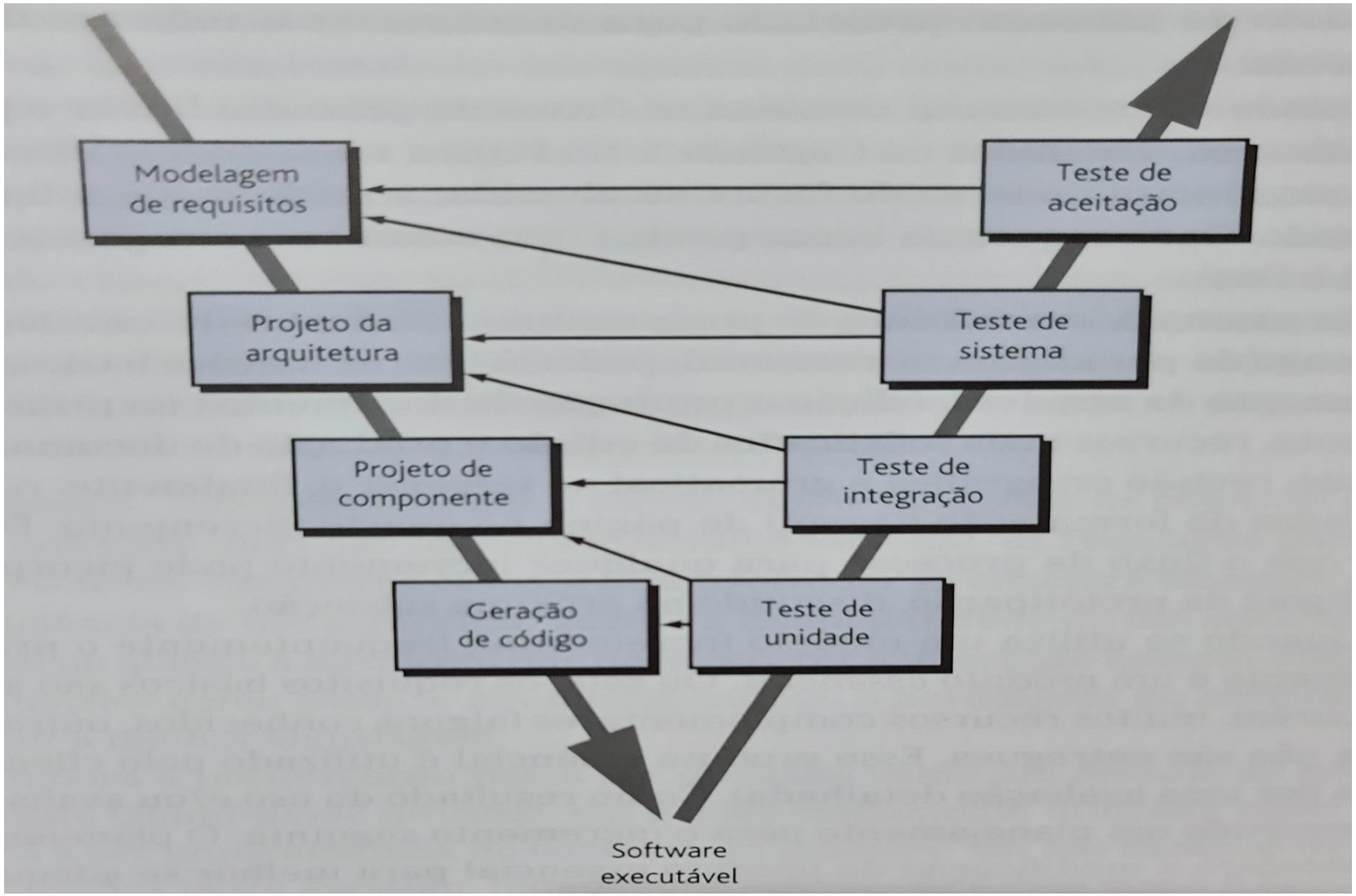
Modelos de processo
1.1 Cascata (ciclo de vida clássico)

Desvantagens

2. Frequentemente, é difícil para o cliente estabelecer explicitamente todas as necessidades. O modelo cascata exige isso e tem dificuldade para adequar a incerteza natural existente no início de muitos projetos.
3. O cliente deve ter paciência. Uma versão operacional do(s) programa(s) não estará disponível antes de estarmos próximos ao final do projeto. Um erro grave, se não detectado até o programa operacional ser revisto, pode ser desastroso.

Modelos de processo

Cascata em V



Modelos de processo

Modelo em V (uma variação do modelo em cascata**)**

Uma variação na representação do modelo cascata é denominada modelo V. O modelo V descreve a relação entre ações de garantia da qualidade e ações associadas a comunicação, modelagem e atividades de construção iniciais. À medida que a equipe de software desce em direção ao lado esquerdo do V, os requisitos básicos do problema são refinados em representações cada vez mais detalhadas e técnicas do problema e de sua solução.

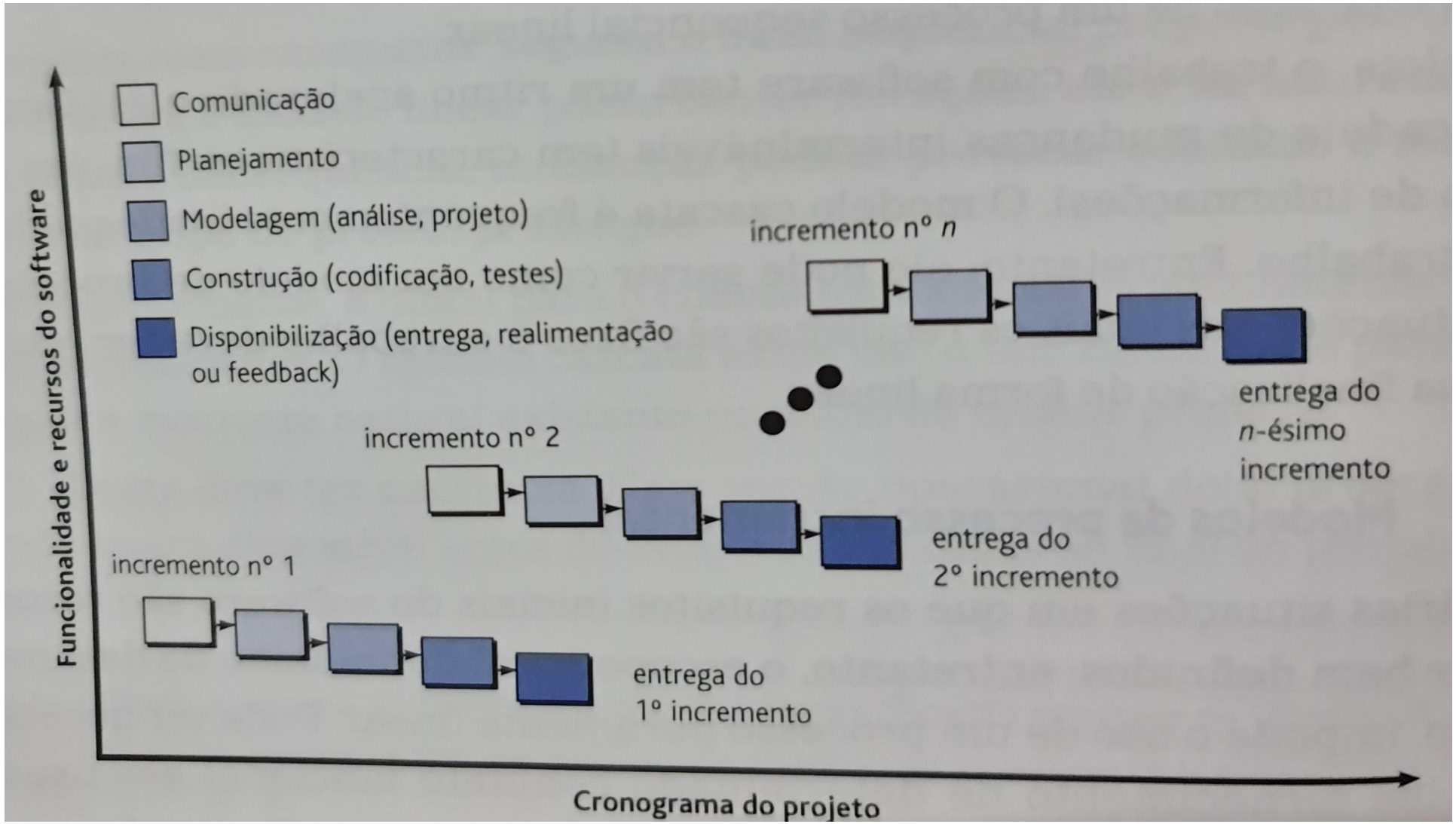
Modelos de processo

Modelo em V (uma variação do modelo em cascata**)**

Uma vez gerado o código, a equipe passa para o lado direito do V, basicamente realizando uma série de testes (ações de garantia da qualidade) que validam cada um dos modelos criados à medida que a equipe desce pelo lado esquerdo. Na realidade, não há nenhuma diferença fundamental entre o ciclo de vida clássico e o modelo V. O modelo V oferece uma maneira de visualizar como as ações de verificação e validação são aplicadas a um trabalho de engenharia anterior.

Modelos de processo

1.2 Incremental



Modelos de processo

1.2 Modelo Incremental

Este modelo libera uma série de versões, denominadas incrementos, que oferecem, progressivamente, maior funcionalidade ao cliente à medida que cada incremento é entregue.

Seu cliente exige a entrega em uma data impossível de atender. Sugira entregar um ou mais incrementos nessa data e o restante do software (incrementos adicionais) posteriormente.

Modelos de processo

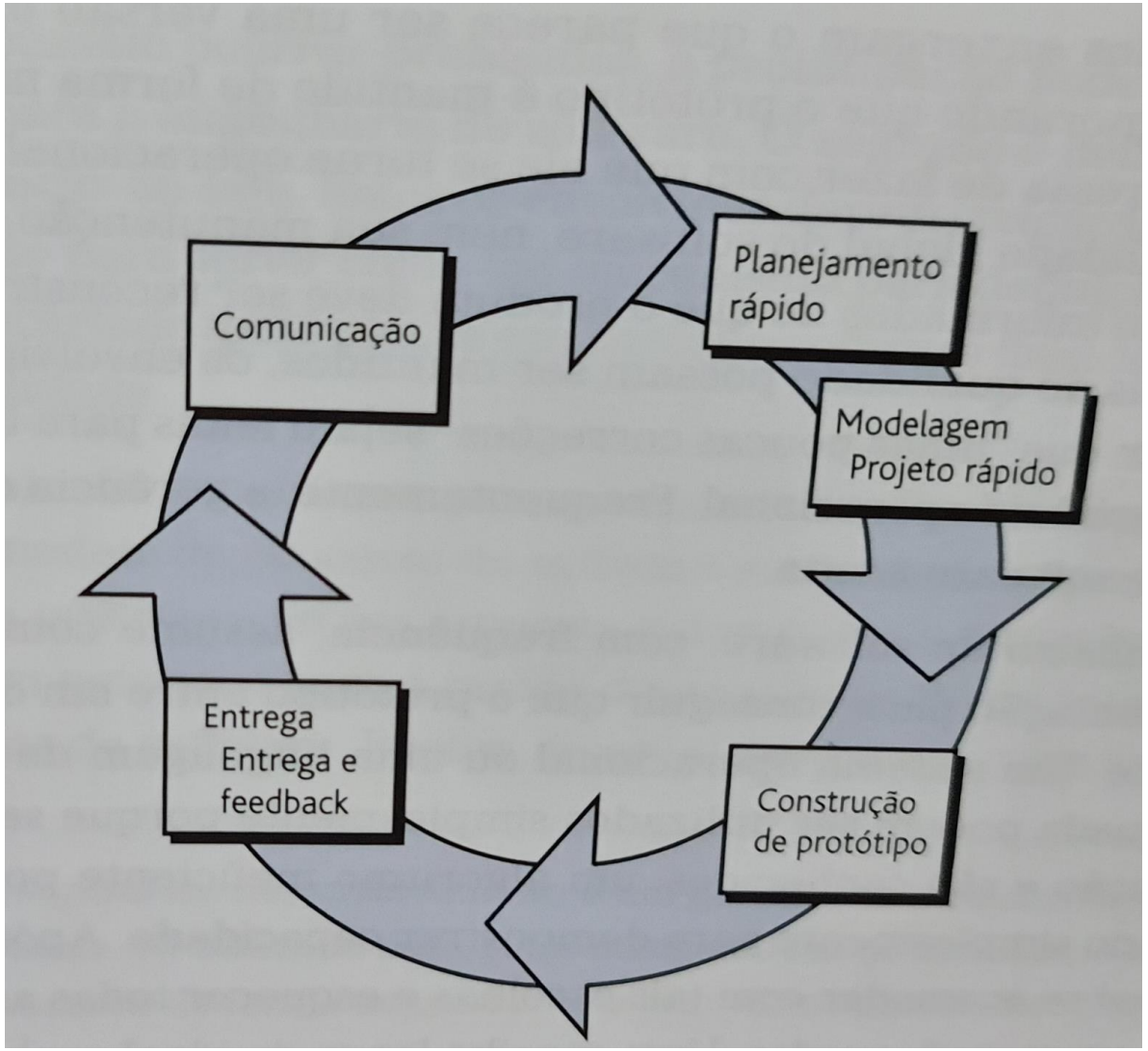
1.3 Modelo de processo evolucionário

Modelos evolucionários são iterativos. Apresentam características que possibilitam desenvolver versões cada vez mais completas do software. A seguir, veremos dois modelos comuns de processos evolucionários:

- Prototipação
- Espiral

Modelos de processo

1.3.1 Prototipação



Modelos de processo

1.3.1 Prototipação

Frequentemente, o cliente define uma série de objetivos gerais o para o software, mas não identifica, detalhadamente, os requisitos para funções e recursos. Em outros casos, o desenvolvedor se encontra inseguro quanto à eficiência de um algoritmo, quanto à adaptabilidade de um sistema operacional ou quanto à forma em que deve ocorrer a interação homem-máquina. Em situações como essas, e em muitas outras, o paradigma da prototipação pode ser a melhor abordagem.

Modelos de processo

1.3.1 Prototipação

Embora a prototipação possa ser utilizada como um modelo de processo isolado (stand-alone process), ela é mais comumente utilizada como uma técnica a ser implementada no contexto de qualquer um dos modelos de processo citados neste capítulo. Independentemente da forma como é aplicado, quando os requisitos estão obscuros, o paradigma da prototipação auxilia os envolvidos a compreender melhor o que está para ser construído.

Modelos de processo

1.3.1 Prototipação

O paradigma da prototipação (Figura acima) começa com a comunicação. Faz-se uma reunião com os envolvidos para definir os objetivos gerais do software, identificar os requisitos já conhecidos e esquematizar quais áreas necessitam, obrigatoriamente, de uma definição mais ampla. Uma iteração de prototipação é planejada rapidamente e ocorre a modelagem (na forma de um “projeto rápido"). Um projeto rápido se concentra em uma representação dos aspectos do software que serão visíveis para os usuários (por exemplo, o layout da interface com o usuário ou os formatos de exibição na tela).

Modelos de processo

1.3.1 Prototipação

O projeto rápido leva à construção de um protótipo. O protótipo é entregue e avaliado pelos envolvidos, os quais fornecem feedback que é usado para refinar ainda mais os requisitos. A iteração ocorre conforme se ajusta o protótipo às necessidades de vários envolvidos e, ao mesmo tempo, possibilita a melhor compreensão das necessidades que devem ser atendidas.

Modelos de processo

1.3.1 Prototipação

Na sua forma ideal, o protótipo atua como um mecanismo para identificar os requisitos do software. Caso seja necessário desenvolver um protótipo operacional, pode-se utilizar partes de programas existentes ou aplicar ferramentas que possibilitem gerar rapidamente tais programas operacionais.

Modelos de processo

1.3.1 Prototipação

O que fazer com o protótipo quando este já serviu ao propósito descrito anteriormente? Na maioria dos projetos, o primeiro sistema dificilmente é útil. Pode ser lento demais, grande demais, estranho em sua utilização ou as três coisas juntas. Não há alternativa, a não ser começar de novo - ressentido, porém mais esperto - e desenvolver uma versão reformulada na qual esses problemas são resolvidos.

Modelos de processo

1.3.1 Prototipação

O protótipo pode servir como “o primeiro sistema”. Aquele recomendado que se jogue fora. Porém, essa pode ser uma visão idealizada. Embora alguns protótipos sejam construídos como “descartáveis”, outros são evolucionários, no sentido de que evoluem lentamente até se transformarem no sistema real.

.

Modelos de processo

1.3.1 Prototipação

Tanto os envolvidos quanto os engenheiros de software gostam do paradigma da prototipação. Os usuários podem ter uma ideia prévia do sistema final, ao passo que os desenvolvedores passam desenvolver algo imediatamente. Entretanto, a prototipação pode ser problemática pelas seguintes razões:

Modelos de processo

1.3.1 Prototipação

1. Os envolvidos enxergam o que parece ser uma versão operacional do software, ignorando que o protótipo é mantido de forma não organizada e que, na pressa de fazer com que ele se torne operacional, não se considera a qualidade global do software, nem sua manutenção em longo prazo. Quando informados de que o produto deve ser reconstruído para que altos níveis de qualidade possam ser mantidos, os envolvidos protestam e solicitam que "umas poucas correções sejam feitas para tornar o protótipo um produto operacional". Frequentemente, a gerência do desenvolvimento de software aceita.

Modelos de processo

1.3.1 Prototipação

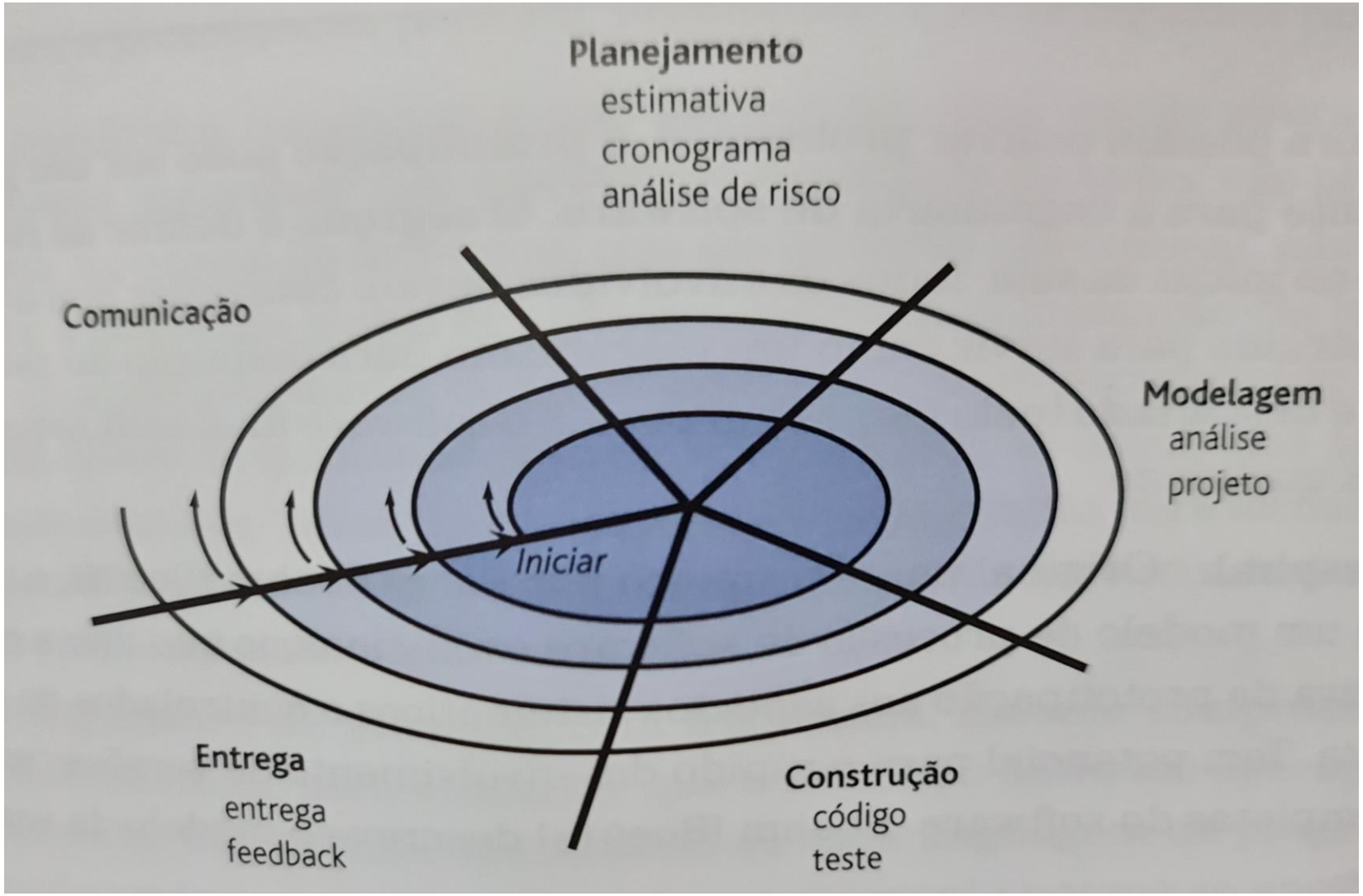
2. O engenheiro de software, com frequência, assume compromissos de implementação para conseguir que o protótipo entre em operação rapidamente. Um sistema operacional ou uma linguagem de programação inadequada podem ser utilizados simplesmente porque se encontram à disposição e são conhecidos; um algoritmo ineficiente pode ser implementado simplesmente para demonstrar capacidade. Após um tempo, é possível se acomodar com tais escolhas e esquecer todas as razões pelas quais eram inadequadas. Uma escolha longe da ideal acaba se tornando parte do sistema.

Modelos de processo

1.3.1 Prototipação

Embora possam ocorrer problemas, a prototipação pode ser um paradigma eficiente para a engenharia de software. O segredo é definir as regras do jogo logo no início: ou seja, todos os envolvidos devem concordar que o protótipo é construído para servir como um mecanismo para definição de requisitos e depois é descartado (pelo menos em parte); o software final será arquitetado visando à qualidade.

Modelos de processo/
1.4 Espiral



Modelos de processo

1.3.2 Espiral

Originalmente proposto por Barry Boehm, o espiral é um modelo de processo de software evolucionário que une a natureza iterativa da prototipação aos aspectos sistemáticos e controlados do modelo cascata. Tem potencial para o rápido desenvolvimento de versões cada vez mais completas do software. Boehm descreve o modelo da maneira:

Modelos de processo

1.3.2 Espiral

- O modelo espiral de desenvolvimento é um gerador de modelos de processos dirigidos a riscos e é utilizado para guiar a engenharia de sistemas com muito software, que ocorre de forma concorrente e tem vários envolvidos. Possui duas características principais que o distinguem.

Modelos de processo

1.3.2 Espiral

- A primeira consiste em uma estratégia cíclica voltada para ampliar, de forma incremental, o grau de definição e a implementação de um sistema, enquanto diminui o grau de risco dele. A segunda característica é que possui uma série de marcos de pontos-âncora para garantir o comprometimento dos envolvidos quanto à busca de soluções de sistema que sejam mutuamente satisfatórias e viáveis.

Modelos de processo

1.3.2 Espiral

Com o modelo espiral, o software será desenvolvido em uma série de versões evolucionárias. Nas primeiras iterações, a versão pode consistir em um modelo ou em um protótipo. Já nas iterações posteriores, são produzidas versões cada vez mais completas do sistema que passa pelo processo de engenharia

Modelos de processo

1.3.2 Espiral

O modelo espiral é dividido em um conjunto de atividades metodológicas definidas pela equipe de engenharia de software. A título de ilustração, utilizam-se as atividades metodológicas genéricas discutidas anteriormente. Cada uma dessas atividades representa um segmento do caminho espiral ilustrado na Figura anterior.

Modelos de processo

1.3.2 Espiral

Assim que esse processo evolucionário começa, a equipe de software realiza atividades indicadas por um circuito em torno da espiral, no sentido horário, começando pelo seu centro. Os riscos são levados em conta à medida que cada revolução é realizada. Marcos de pontos-âncora -uma combinação de artefatos e condições satisfeitas ao longo do trajeto da espiral - são indicados para cada passagem evolucionária.

Modelos de processo

1.3.2 Espiral

O primeiro circuito em volta da espiral pode resultar no desenvolvimento de uma especificação de produto; passagens subsequentes em torno da espiral podem ser usadas para desenvolver um protótipo e, então, progressivamente, versões cada vez mais sofisticadas do software. Cada passagem pela região de planejamento resulta em ajustes no planejamento do projeto. Custo e cronograma são ajustados de acordo com o feedback (a realimentação) obtido do cliente após a entrega. Além disso, o gerente de projeto faz um ajuste no número de iterações planejadas para concluir o software.

Modelos de processo

1.3.2 Espiral

Diferentemente de outros modelos de processo, que terminam quando o software é entregue, o modelo espiral pode ser adaptado para ser aplicado ao longo da vida do software. Logo, o primeiro circuito em torno da espiral pode representar um “projeto de desenvolvimento de conceitos” que começa no núcleo da espiral e continua por várias iterações até que o desenvolvimento de conceitos esteja concluído. Se o conceito for desenvolvido para ser um produto final, o processo prossegue na espiral pelas “bordas" e um “novo projeto de desenvolvimento de produto" se inicia. O novo produto evoluirá, passando por iterações em torno da espiral.

Modelos de processo

1.3.2 Espiral

Mais tarde, uma volta em torno da espiral pode ser usada para representar um projeto de aperfeiçoamento do produto". Basicamente, a espiral, quando caracterizada dessa maneira, permanece em operação até que o software seja retirado. Há casos em que o processo fica inativo; porém, toda vez que uma mudança é iniciada, começa no ponto de partida apropriado (por exemplo, aperfeiçoamento do produto).

Modelos de processo

1.3.2 Espiral

O modelo espiral é uma abordagem realista para o desenvolvimento de sistemas e de software em larga escala. Como o software evolui à medida que o processo avança, o desenvolvedor e o cliente compreendem e reagem melhor aos riscos em cada nível evolucionário. Esse modelo usa a prototipação como mecanismo de redução de riscos e, mais importante, torna possível a aplicação da prototipação em qualquer estágio do processo evolutivo do produto.

Modelos de processo

1.3.2 Espiral

Ele mantém a abordagem em etapas, de forma sistemática, sugerida pelo ciclo de vida clássico, mas a incorpora em uma metodologia iterativa que reflete mais realisticamente o mundo real. O modelo espiral exige consideração direta dos riscos técnicos em todos os estágios do projeto e, se aplicado apropriadamente, reduz os riscos antes de se tornarem problemáticos.

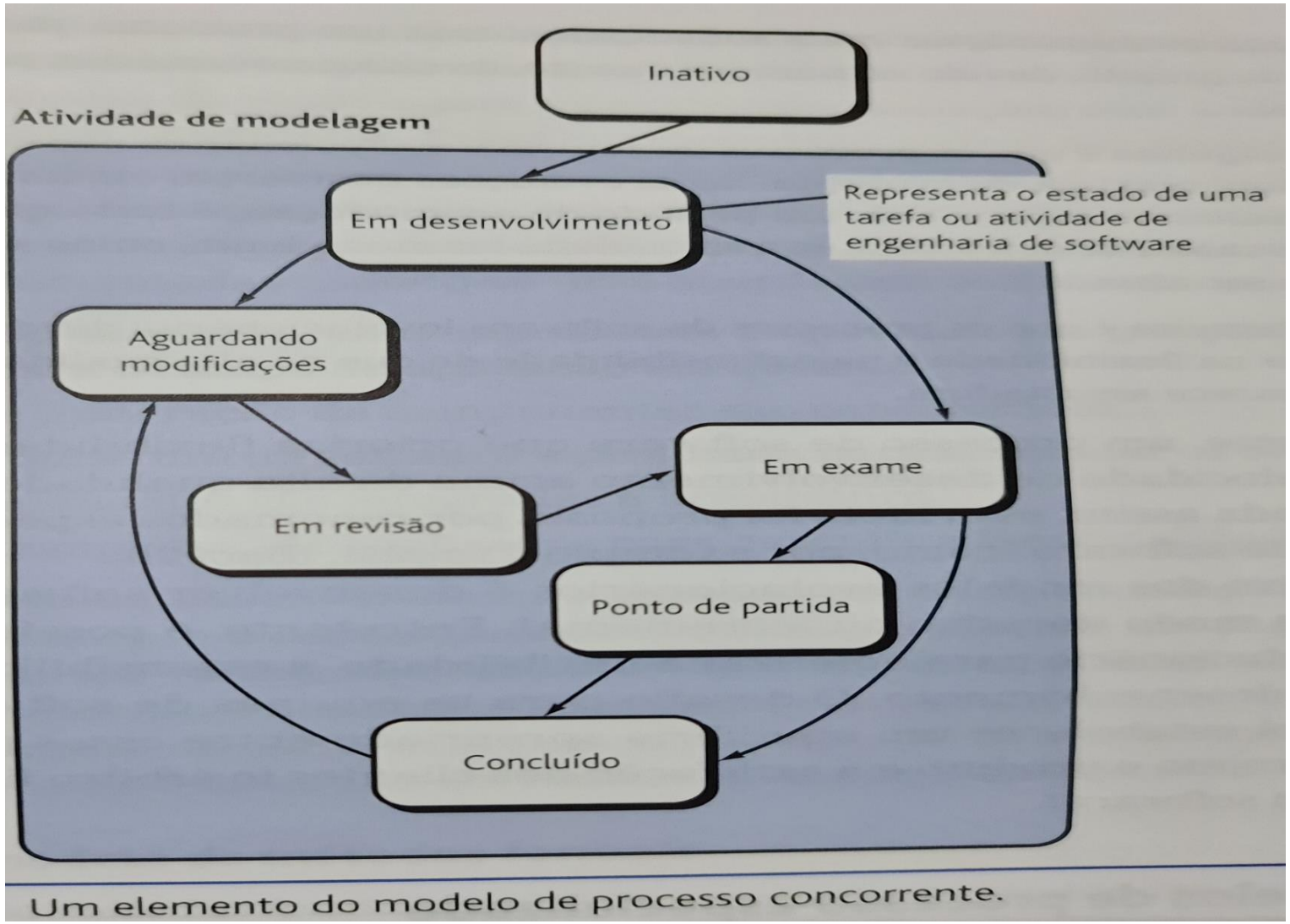
Modelos de processo

1.3.2 Espiral

Como outros paradigmas, esse modelo não é uma panaceia. Pode ser difícil convencer os clientes (particularmente em situações contratuais) de que a abordagem evolucionária é controlável. Ela exige considerável especialização na avaliação de riscos e depende dessa especialização para seu sucesso. Se um risco muito importante não for descoberto e administrado, sem dúvida ocorrerão problemas.

Modelos de processo

1.5 Concorrente



Um elemento do modelo de processo concorrente.

Modelos de processo

1.5 Concorrente

O modelo de desenvolvimento concorrente, por vezes chamado de engenharia concorrente, possibilita à equipe de software representar elementos concorrentes e iterativos de qualquer um dos modelos de processo descritos neste capítulo. Por exemplo, a atividade de modelagem definida para o modelo espiral é realizada invocando uma ou mais destas ações de engenharia de software: prototipação, análise e projeto.

Modelos de processo

1.5 Concorrente

A Figura acima mostra um exemplo de abordagem de modelagem concorrente. Uma atividade - modelagem - poderia estar em qualquer dos estados observados em qualquer momento determinado. Similarmente, outras atividades, ações ou tarefas (por exemplo, comunicação ou construção) podem ser representadas de maneira análoga. Todas as atividades de engenharia de software existem simultaneamente, porém estão em diferentes estados.

Modelos de processo

1.5 Concorrente

Por exemplo, no início de um projeto, a atividade de comunicação (não mostrada na figura) completou sua primeira iteração e se encontra no estado aguardando modificações. A atividade de modelagem (que se encontrava no estado nenhum, enquanto a comunicação inicial era concluída) agora faz uma transição para o estado em desenvolvimento. Entretanto, se o cliente indicar que devem ser feitas mudanças nos requisitos, a atividade de modelagem passa do estado em desenvolvimento para o estado aguardando modificações.

Modelos de processo

1.5 Concorrente

A modelagem concorrente define uma série de eventos que vão disparar transições de um estado para outro para cada uma das atividades, ações ou tarefas da engenharia de software. Por exemplo, durante os estágios iniciais do projeto (uma ação de engenharia de software importante que ocorre durante a atividade de modelagem), uma inconsistência no modelo de requisitos não é descoberta. Isso gera o evento correção do modelo de análise, que vai disparar a ação de análise de requisitos, passando do estado concluído para o estado aguardando modificações.

Modelos de processo

1.5 Concorrente

A modelagem concorrente se aplica a todos os tipos de desenvolvimento de software e fornece uma imagem precisa do estado atual de um projeto.

Em vez de limitar as atividades, ações e tarefas da engenharia de software a uma sequência de eventos, ela define uma rede de processos. Cada atividade, ação ou tarefa na rede existe simultaneamente com outras atividades, ações ou tarefas. Eventos gerados em um ponto da rede de processos dispararam transições entre os associados a cada atividade.

Modelos de processo

1.5 Concorrente

A modelagem concorrente se aplica a todos os tipos de desenvolvimento de software e fornece uma imagem precisa do estado atual de um projeto.

Em vez de limitar as atividades, ações e tarefas da engenharia de software a uma sequência de eventos, ela define uma rede de processos. Cada atividade, ação ou tarefa na rede existe simultaneamente com outras atividades, ações ou tarefas. Eventos gerados em um ponto da rede de processos dispararam transições entre os associados a cada atividade.

Modelos de processo

Os quatro pilares da Orientação a Objetos

- **Abstração**

  É a habilidade de concentrar nos aspectos essenciais de um contexto qualquer, ignorando características menos importantes ou acidentais. Em modelagem **orientada a objetos**, uma classe é uma **abstração** de entidades existentes no domínio do sistema de software.

Modelos de processo

Os quatro pilares da Orientação a Objetos

- **Herança**

  Para fazer uma analogia próxima à realidade não virtual, em uma família, por exemplo, a criança herda diretamente do pai e indiretamente do avô e do bisavô. Em programação, a lógica é similar. Assim, os objetos filhos herdam as características e ações de seus ancestrais”.

Modelos de processo

Os quatro pilares da Orientação a Objetos

- **Encapsulamento**

  O encapsulamento é uma técnica que adiciona segurança à aplicação em uma programação orientada a objetos, pois esconde as propriedades, criando uma espécie de caixa preta.

  Muitas das linguagens orientadas a objetos implementam o encapsulamento baseado em propriedades privadas, por métodos chamados getters e setters, responsáveis por retornar e gravar o valor da propriedade, respectivamente. Assim, se evita o acesso direto ao atributo do objeto, adicionando outra camada de segurança à aplicação.

Modelos de processo

Os quatro pilares da Orientação a Objetos

- **Poliformismo**

Na natureza, existem animais que são capazes de alterar sua forma conforme a necessidade. Na orientação a objetos a ideia é a mesma.

O poliformismo permite herdar um método de classe pai e atribuir uma nova implementação para o método pré-definido.

Modelos de processo

UML (*Unified Modeling Language* – Linguagem de Modelagem Unificada).

- Evolução do paradigma de orientação a objetos na década de 90;
- Existiam diversas propostas de métodos e análise de OO.
- Nasce a UML da união das melhores características de seus modelos de desenvolvimento de software, através dos pesquisadores – James Rumbaugh, Grady Booch e Ivar Jacobson;
- O Objetivo era construir um modelo que pudesse se tornar um padrão de referência para modelagem orientada a objeto.

Modelos de processo

UML (*Unified Modeling Language* – Linguagem de Modelagem Unificada).

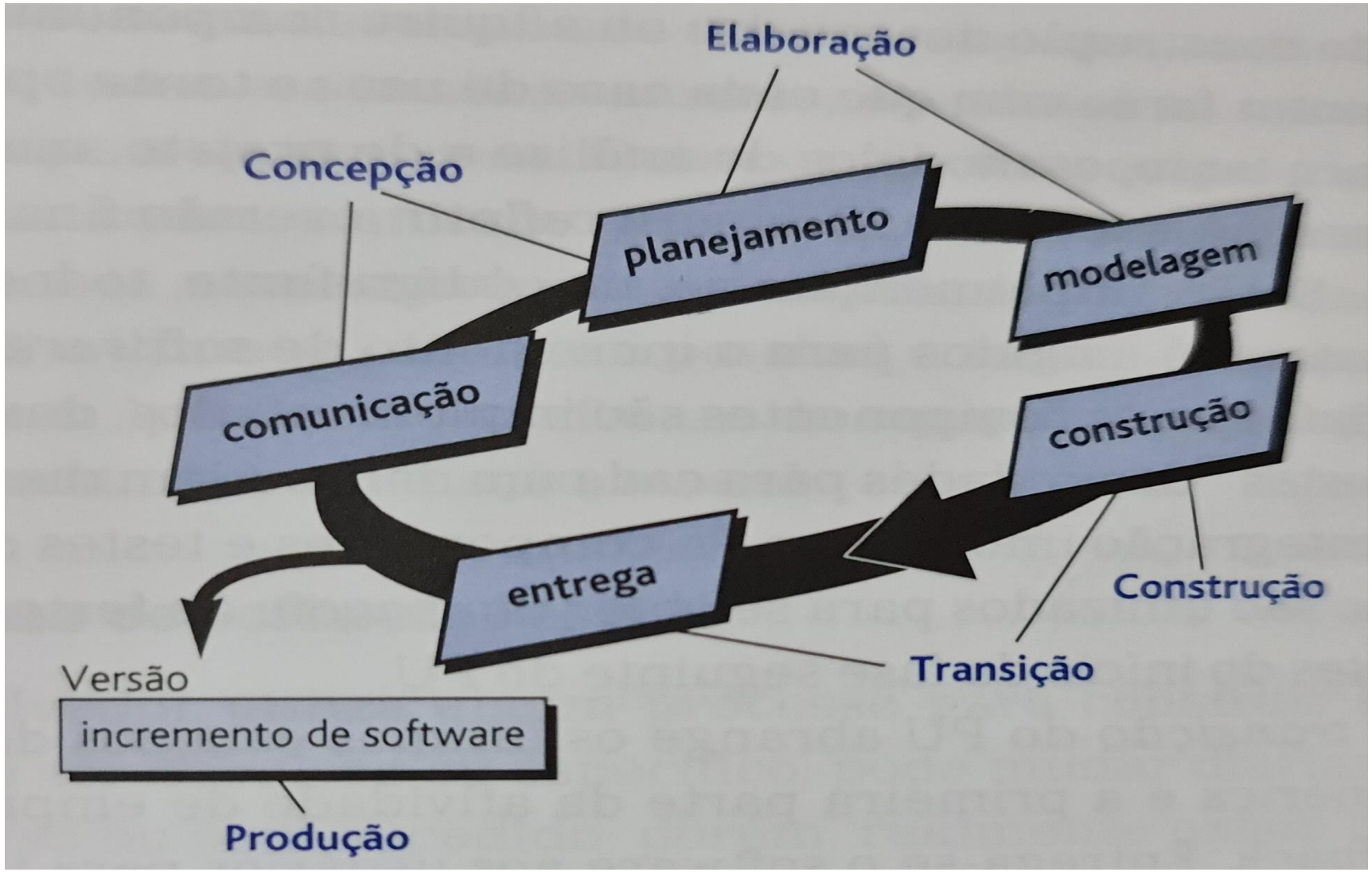
- Podemos dizer que a UML se encaixa como um método.
- Suporta diversas ferramentas automatizadas
- Criou-se então o seguinte contexto: As pessoas tinham um método disponível para modelagem, bem como diversas ferramentas automatizadas, mas não tinham um processo para guiar o desenvolvimento.

Modelos de processo
Processo Unificado

- Os mesmos pesquisadores que propuseram a UML, criaram o Processo Unificado.
- Permite uma maior adaptação a realidade e as constantes mudanças nos requisitos;
- Existe uma grande quantidade de projetos desenvolvidos usando o Processo Unificado, ou adaptações deste processo.
- O RUP é uma das versões comerciais do Processo Unificado, e foi criado pela Rational Software Corporation, adquirida pela IBM. EUP (Enterprise Unified Process), OpenUp e OpenUp/Basic também são variações dele.

Modelos de processo

2.1 Processo Unificado



Modelos de processo
Processo Unificado

- O principal objetivo do UP é atender as necessidades dos usuários garantindo uma produção de software de alta qualidade que cumpra um cronograma e um orçamento previsíveis. Assim, o UP mostra como o sistema será construído na fase de implementação, gerando o modelo do projeto e, opcionalmente, o modelo de análise que é utilizado para garantir a robustez. O UP define perfeitamente quem é responsável pelo que, como as coisas deverão ser feitas e quando devem ser realizadas, descrevendo todas as metas de desenvolvimento especificamente para que sejam alcançadas.

27/10mModelos de processo

Processo Unificado

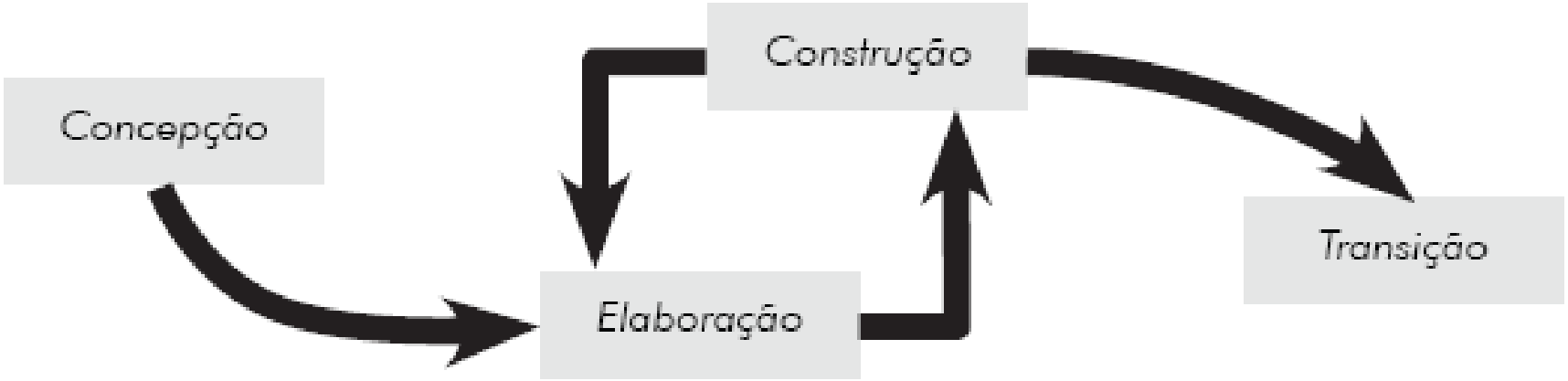
- O RUP organiza o desenvolvimento de software em quatro fases, onde são tratadas questões sobre planejamento, levantamento de requisitos, análise, implementação, teste e implantação do software. Cada fase tem um papel fundamental para que o objetivo seja cumprido, distribuídos entre vários profissionais como o Analista de sistema, Projetista, Projetista de testes, entre outros.

Modelos de processo
Processo Unificado

- Fases do processo unificado também se encaixam nas fases do arcabouço genérico, com quatro fases distintas:
  - Concepção;
  - Elaboração;
  - Construção;
  - Transição.

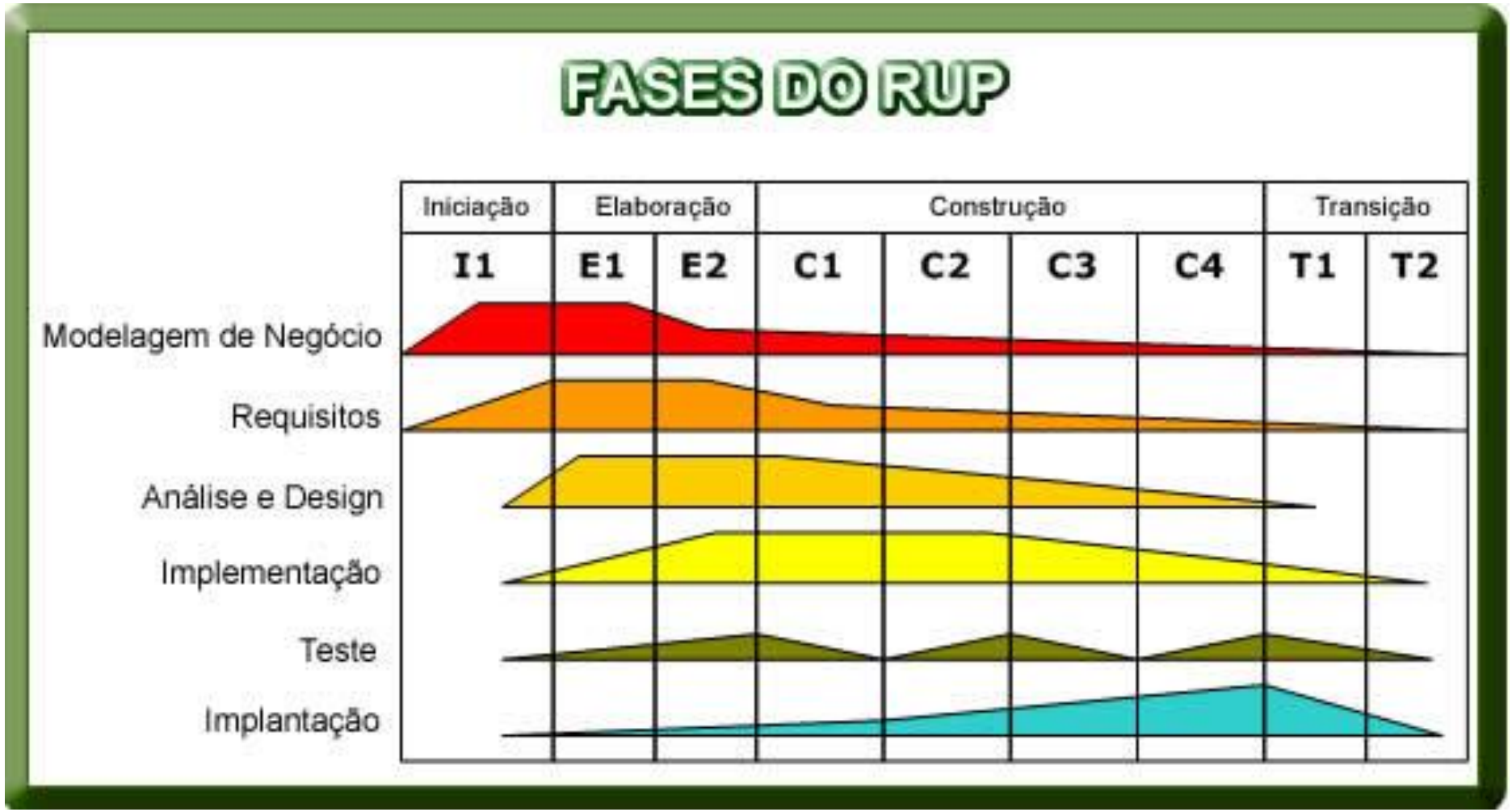
Modelos de processo

Processo Unificado



Modelos de processo

Processo Unificado



Modelos de processo
Processo Unificado

- **Fase de Concepção / Iniciação**: Esta fase do RUP abrange as tarefas de comunicação com o cliente e planejamento. É feito um plano de projeto avaliando os possíveis riscos, as estimativas de custo e prazos, estabelecendo as prioridades, levantamento dos requisitos do sistema e preliminarmente analisá-lo. Assim, haverá uma anuência das partes interessadas na definição do escopo do projeto, onde são examinados os objetivos para se decidir sobre a continuidade do desenvolvimento. Coleção inicial de casos de uso.
- É feito um estudo de viabilidade e são identificados os requisitos para o software, que são obtidos em uma forma mais genérica.

Modelos de processo
Processo Unificado

- **Fase de Elaboração:** Abrange a Modelagem do modelo genérico do processo. O objetivo desta fase é analisar de forma mais detalhada a análise do domínio do problema, revisando os riscos que o projeto pode sofrer e a arquitetura do projeto começa a ter sua forma básica. Indagações como "O plano do projeto é confiável?", "Os custos são admissíveis?" são esclarecidas nesta etapa.
- Um dos principais objetivos desta etapa é criar um conjunto de classes que descreva o comportamento do sistema.

Modelos de processo
Processo Unificado

- **Fase de Construção:** Desenvolve ou Adquire os componentes de Software. O principal objetivo desta fase é a construção do sistema de software, com foco no desenvolvimento de componentes e outros recursos do sistema. É na fase de Construção que a maior parte de codificação ocorre.
- São projetados e executados testes unitários (um teste unitário tem por objetivo testar cada funcionalidade do sistema separadamente);
- Também é feita a integração da funcionalidade produzida com as funcionalidades já existentes;
- Falhas ou possíveis melhorias são automaticamente reportados fazendo com que o projeto volte a sua elaboração.

Modelos de processo
Processo Unificado

- **Fase de Transição:** Ocorre somente ao final do projeto, após o último ciclo iterativo. Abrange a entrega do software ao usuário. O software é testado no ambiente do cliente, que dá os retornos sobre as modificações necessárias. O objetivo desta fase é disponibilizar o sistema, tornando-o disponível e compreendido pelo usuário final. As atividades desta fase incluem o treinamento dos usuários finais e também a realização de testes da versão beta do sistema visando garantir que o mesmo possua o nível adequado de qualidade,
- Também são criadas informações de apoio, como manuais, guias de solução de problemas, procedimentos de instalação.

Modelos de processo
Processo Unificado

Ciclo Iterativo: Entre elaboração e construção.

O trabalho maior de análise, projeto e implementação é realizado em ciclos curtos.

Quando estes ciclos iterativos terminam:

- tem-se o sistema praticamente pronto;
- faltando apenas os testes finais de integração;
- implantação junto ao usuário.
- Podemos dizer que, o Processo Unificado também é iterativo;
- É necessário que o cliente esteja envolvido no processo e esteja a par das funcionalidades desenvolvidas desde o início do processo de desenvolvimento.

Bibliografia

Pressman, Roger S. Engenharia de Software: uma abordagem profissional/Roger S. Pressman, Bruce R. Maxim. 8ª Edição Porto Alegre: AMGH, 2016

Martin, James e McClure Carma Técnicas Estruturadas e Case. São Paulo, Makron, 1991

Fournier, Roger Guia prático para desenvolvimento e manutenção de sistemas estruturados, São Paulo, Makron, 1994