Analise e Modelagem de Sistemas

Modelos de processos de software

Eduardo Furlan Miranda

2025-07-01

Fonte: WERLICH, C. *Análise e Modelagem de Sistemas*. Londrina: EDE SA, 2020. ISBN 978-85-522-1683-4.

O Desafio do Analista de Sistemas

- Cenário atual: desenvolvimento de APPs nativos para Android e iOS exige diferentes linguagens
  - Necessita de dois projetos e duas equipes para um mesmo APP nativo
  - Requer maneira organizada, controlada e eficiente de trabalho
- Nova missão: cliente deseja Software para loja de brinquedos online
  - Software para site e aplicativo para compra/aluguel de brinquedos
- Seu objetivo: determinar o Modelo de Processo de Software mais indicado para o projeto

O Que É um Processo de Software?

- Definição: conjunto de atividades cuja finalidade é ter como resultado um Produto de Software
  - Envolve muitas tarefas distribuídas entre os membros da equipe
- Atividades destaques (Sommerville 2011):
  - Estudo da viabilidade e análise de requisitos
  - Especificação, arquitetura de software e implementação (codificação)
  - Testes, documentação, suporte, treinamento e manutenção

Atividades Genéricas de um Processo de Software

- Atividades essenciais (Sommerville 2011):
  - Análise e especificação: definições do software, requisitos e restrições
  - Projeto: alocação de recursos (hardware e software), identificação de abstrações
- Fases de construção e evolução:
  - Implementação e teste unitário: codificação e fabricação do software
  - Integração e verificação: avaliação, correção e validação da qualidade final
  - Operação e manutenção: processos de alterações pós-funcionamento

Modelos de Processos de Software: Propósito

- Objetivo: propiciar estabilidade, controle e organização das atividades
  - Representação (geralmente gráfica) de objetos e atividades
- Guia exclusivo (Pressman 2016):
  - Define um fluxo de todas as atividades, ações e tarefas
  - Determina o nível de interação entre atividades e artefatos produzidos
- Adaptação: cada empresa adota seu modelo, realizando adaptações a cada software
  - Não há consenso sobre o melhor modelo (Pfleeger 2004)

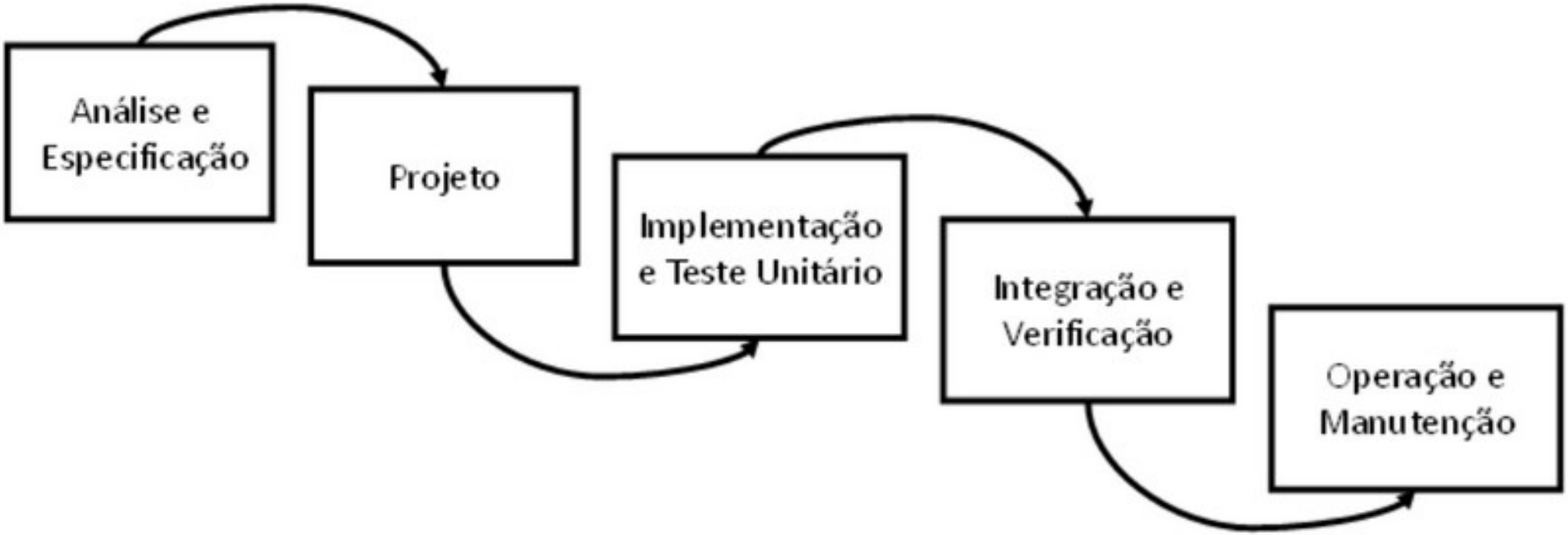
Categorias de Modelos de Processos de Software

- Diversidade: existem diversos Modelos que possuem características diferentes
- Finalidade comum: evitar o caos no desenvolvimento e estabelecer um Fluxo de Trabalho
- Principais categorias:
  - Modelos de Processos Prescritivos
  - Modelos de Processos Especializados
  - Modelos de Desenvolvimento Ágil

Modelo de Processo Prescritivo

- Definição: conjunto de elementos do processo (Pressman 2016)
  - Conhecido como Modelos de Processos Tradicionais
  - Inclui ações de engenharia de software, produtos de trabalho
  - Mecanismos que garantem qualidade e controle de mudanças
- Estruturação: prescreve os relacionamentos dos elementos para estruturar o desenvolvimento
- Diretrizes definidas (Pressman 2016):
  - Tarefas ocorrem de forma sequencial
  - Indica série de atividades metodológicas, ações, tarefas, artefatos
  - Garantias de qualidade e mecanismos de controle de mudanças

Modelo de Processo Prescritivo: Modelo Cascata



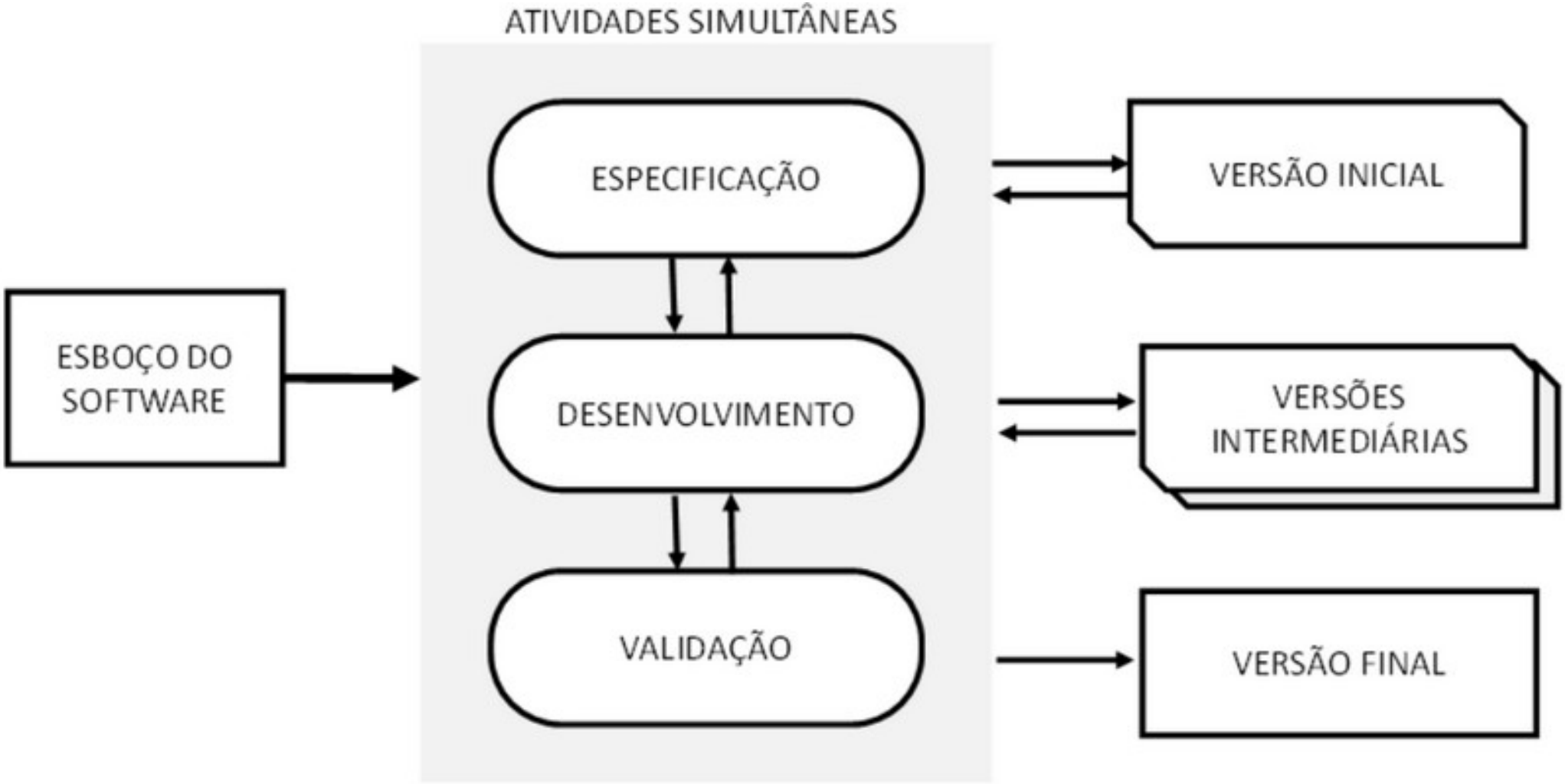
Modelo Cascata: Visão Geral

- Enfoque: sistemático e sequencial dos Processos
  - Cada fase é iniciada somente após a conclusão da fase anterior
- Antiguidade e uso: um dos mais antigos e ainda utilizados (Pressman 2016)
- Estágios fundamentais (Sommerville 2011):
  - Análise e Especificação
  - Projeto
  - Implementação e Teste de Unidade
  - Integração e Verificação
  - Operação e Manutenção

Modelo Cascata: Atividades Detalhadas e Desafios

- Análise e especificação: estabelecimento de funcionalidades e documentação com cliente
- Projeto: definição da estrutura de dados, arquitetura, interfaces gráficas
- Implementação e teste de unidade: codificação e testes em cada módulo
- Integração e verificação: integração e testes de todas as partes para entrega
- Operação e manutenção: correções e novos requisitos após a utilização
- Desvantagens:
  - Desenvolvimento pode se estender por meses, dependendo da complexidade
  - Atraso em uma etapa reflete nas demais
  - Gera insatisfação do cliente devido ao tempo de espera e única fase de requisitos

Modelo de Processo Prescritivo: Modelo Incremental



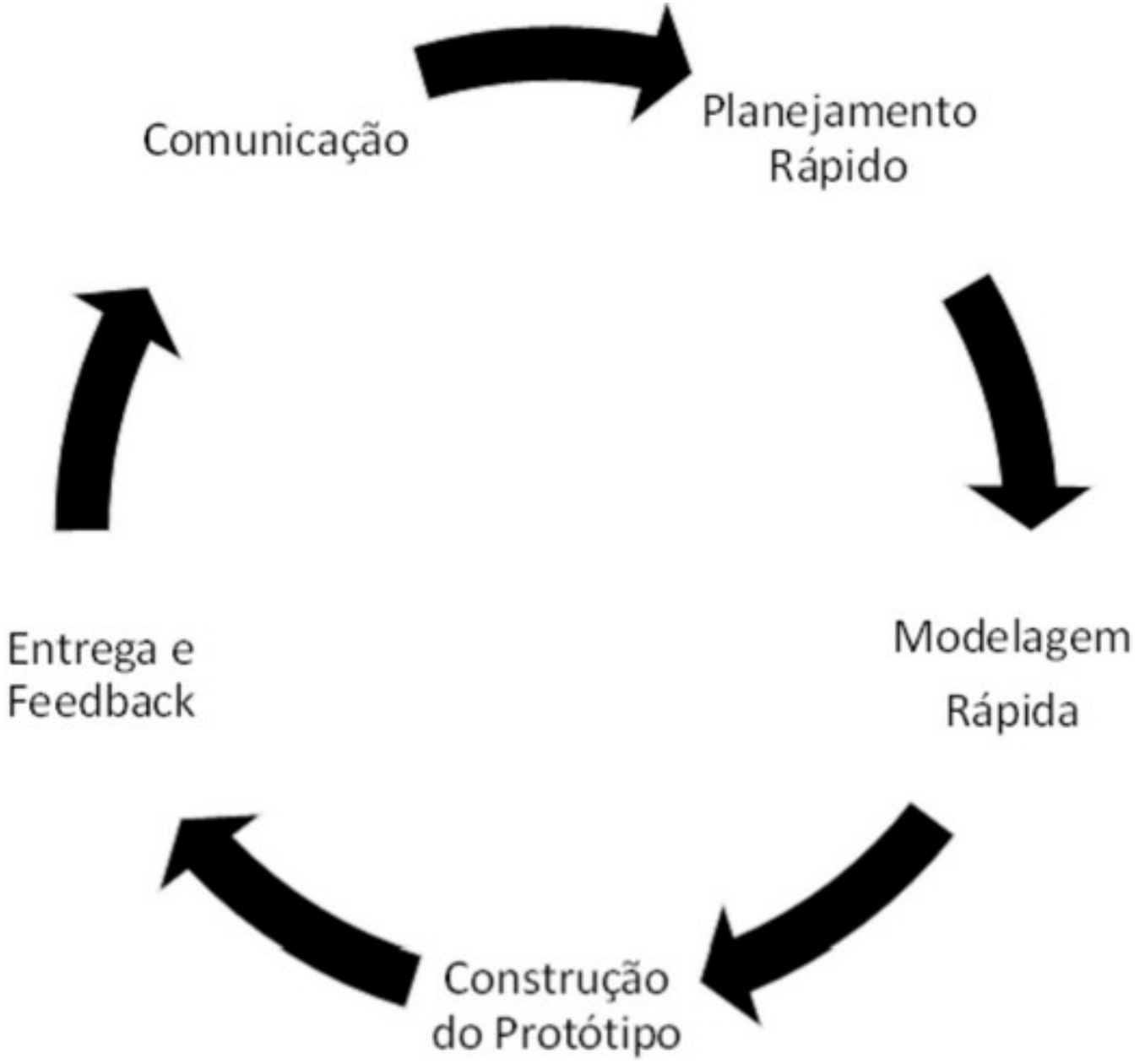
Modelo Incremental

- Definição: modelo iterativo que cria pequenas versões do Software (Sommerville 2011)
  - Software é expandido em novas versões até ser totalmente construído
- Incrementos: uma versão é um incremento
  - Cada incremento incorpora parte da funcionalidade requisitada
  - Cliente recebe "pedaços" do software (módulos que acrescentam ou melhoram)
- Atividades simultâneas:
  - Especificação, Desenvolvimento e Validação são realizadas de forma intercalada
  - Rápido feedback entre as atividades
  - Cada versão produzida é um Sistema funcional, mesmo que incompleto

Modelos Evolucionários

- Definição: produzem uma versão cada vez mais completa do Software
  - Iterativos e evoluem ao longo do tempo (Pressman 2016)
- Adaptação a mudanças: alinham-se a projetos com requisitos de negócio e produto instáveis
  - Software pode ser desenvolvido pensando nesta evolução
- Dois tipos principais:
  - Prototipação
  - Espiral

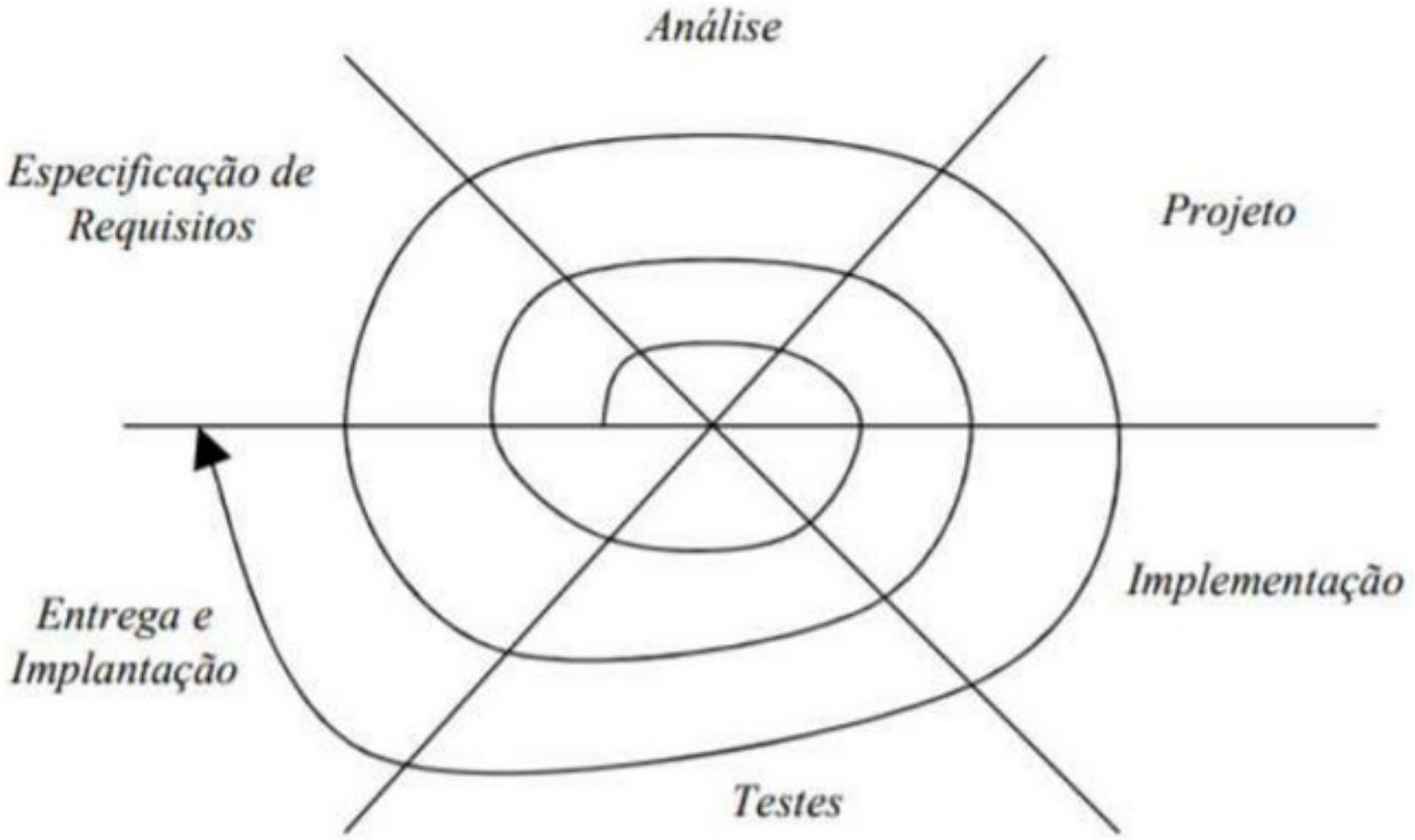
Modelo de Processo Prescritivo: Modelo Evolucionário - Prototipação



Modelo Evolucionário: Prototipação

- Início: definição dos requisitos do Software pelo cliente e Analista de Sistemas
- Ciclo de atividades (Pressman 2016):
  - Comunicação: identificação de objetivos e funcionalidades
  - Planejamento rápido: definição de requisitos para Modelagem e Construção
  - Modelagem rápida e construção do protótipo
  - Entrega e feedback: cliente avalia e a equipe aprimora
- Utilidade: apresenta versão inicial para experimentações e testes
  - Ajuda no entendimento do que será construído para o cliente
  - O protótipo pode se tornar o produto final

Modelo de Processo Prescritivo: Modelo Evolucionário - Espiral



Modelo Evolucionário: Espiral

- Características: iterativo como prototipação
  - Utiliza aspectos sistemáticos e controlados do Modelo Cascata (Pressman 2016)
- Objetivo: fornecer um rápido desenvolvimento de versão
  - A cada ciclo, gera versões mais completas
- Fases comuns:
  - Análise
  - Projeto
  - Implementação
  - Testes
  - Entrega e Implantação

Modelos de Processos Prescritivos Concorrentes

- Representação: graficamente, série de tarefas, técnicas maiores e estados associados
  - Utilizados como paradigma para desenvolvimento de aplicações Cliente/Servidor
- Flexibilidade: permite representar elementos concorrentes e iterativos
  - Integra diferentes tipos de Modelos de Processos de Software
- Aplicações:
  - Utilizados em projetos que envolvem diferentes equipes de desenvolvimento
  - Não segue sequência de atividades, mas estabelece uma rede de atividades
  - Planos de projeto são documentos vivos, avaliados e revisados com frequência

Modelo de Processo Especializado: Introdução

- Base: utilizam muitas das características de um ou mais Modelos Prescritivos
- Necessidade: usados quando existe a necessidade de uma abordagem mais especializada
- Tipos (Pressman 2016):
  - Modelo Baseado em Componentes
  - Modelo de Métodos Formais
  - Desenvolvimento de Software Orientado a Aspecto
  - Modelo de Processo Unificado
  - Modelos de Processos Pessoal e de Equipe

Modelos Especializados: Baseado em Componentes e Métodos Formais

- Modelo baseado em componentes:
  - Utilizados em projetos de Software de Prateleira (Software de Linha)
  - Compreende aplicações de componentes de Software pré-empacotados
  - Componente é parte independente, reutilizável, pode ser trocado ou alterado

- Modelo de métodos formais:
  - Conjunto de atividades que levam à especificação matemática formal do software
  - Fornece mecanismos para descoberta e eliminação de problemas (ambiguidade, inconsistência)
  - Base para averiguação do código de programação e descoberta de erros

Modelos Especializados: Orientado a Aspecto e Unificado

- Desenvolvimento de Software Orientado a Aspecto:
  - Fornece processo e abordagem metodológica para definir, especificar, projetar e construir aspectos
  - Código do software separado de acordo com sua importância (ex: classes OO)
  - Requisitos modelados transcendendo várias funcionalidades do Sistema

- Modelo de Processo Unificado (RUP - Rational Unified Process):
  - Aproveita características dos Modelos de Processos tradicionais (Prescritivos)
  - Implementa alguns princípios da metodologia Ágil
  - Considerado um Modelo iterativo e incremental

Modelos Especializados: Pessoal e de Equipe

- Foco principal: o cerne do desenvolvimento de software está na equipe
- Modelo de processo de software pessoal:
  - Enfatiza a medição pessoal do que foi produzido
  - Avalia o artefato gerado e a qualidade resultante
- Modelo de processo de software de equipe:
  - Objetiva a criação de uma equipe autodirigida e auto-organizada
  - Finalidade: produzir um Software com alto padrão de qualidade

Modelo de Desenvolvimento Ágil: Contexto e Benefícios

- Cenário: rápido surgimento de novas tecnologias e velocidade nos negócios
  - Demanda maior velocidade no desenvolvimento de software
- Propósito: nova forma de desenvolver software, mais flexível e dinâmico
  - Procura resolver problemas da Engenharia de Software
- Resposta rápida (Pressman 2016):
  - Clientes querem resultados rapidamente
  - Novas funcionalidades agregadas conforme ideias ou necessidades
  - Torna o desenvolvimento mais flexível e rápido

Princípios do Desenvolvimento Ágil (Parte 1)

- Foco principal (Pressman 2016): focado nas entregas
  - Prioriza comunicação ativa e contínua
  - Busca a satisfação do cliente através de entregas incrementais
- Envolvimento do cliente (Sommerville 2011):
  - Cliente deve ter forte envolvimento no processo
  - Fornece requisitos e avalia o desenvolvido
- Entrega incremental (Sommerville 2011):
  - Software desenvolvido com incrementos
  - Cliente indica novos requisitos a serem acrescidos

Princípios do Desenvolvimento Ágil (Parte 2)

- Pessoas e não processos (Sommerville 2011):
  - Equipe possui liberdade de desenvolvimento
  - Explora ao máximo a capacidade dos desenvolvedores (não presos a Prescritivos)
- Aceitar as mudanças (Sommerville 2011):
  - Requisitos podem ser alterados
  - Projeto deve ser elaborado pensando nesta possibilidade
- Manter a simplicidade (Sommerville 2011):
  - Complexidade deve ser banida, equipe deve trabalhar de forma simples e ativa
- Redução de burocracia (Pressman 2016):
  - Visa reduzir drasticamente a documentação
  - Torna o processo flexível e reduz a burocracia

Métodos Ágeis em Destaque

- Base comum: todos os métodos ágeis são baseados no Manifesto Ágil
  - Enfatizam colaboração humana e auto-organização
- Métodos mais destacados:
  - XP (Extreme Programming)
  - Scrum
- Outros métodos (Pressman 2016):
  - Método de desenvolvimento de sistemas dinâmicos (DSDM)
  - Modelagem Ágil
  - Processo Unificado Ágil

Método Ágil: XP (Extreme Programming)

- Características principais:
  - Feedback constante
  - Abordagem de desenvolvimento incremental
  - Comunicação primordial entre os envolvidos
- Quatro atividades metodológicas (Pressman 2016):
  - Planejamento
  - Projeto
  - Codificação
  - Testes
- Práticas:
  - Desenvolvimento do software padronizado
  - Trabalho realizado em pares
  - Cliente (representante) sempre presente para esclarecer dúvidas

Método Ágil: Scrum

- Processo: iterativo e incremental
  - Pode ser utilizado em processos gerenciais
- Regras e práticas de gestão:
  - Determina conjunto de regras e práticas para o sucesso dos projetos
  - Exemplos: trabalho em equipe e comunicação melhorada
- Atividades metodológicas (Pressman 2016):
  - Requisitos
  - Análise
  - Projeto
  - Evolução
  - Entrega

Scrum: Backlog e Sprints

- Backlog (Product Backlog):
  - Lista com prioridades dos requisitos (funcionalidades) do projeto
  - Item pode ser adicionado ou eliminado a qualquer momento (alterações)
  - Gerente do produto deve registrar e atualizar as prioridades
  - Conjunto de "Histórias" (ideias e funcionalidades iniciais)
- Sprints:
  - Unidades de trabalho para atingir um requisito do Backlog
  - Precisa ser ajustado dentro do Time Box (Janela de Tempo) para prazos de entrega
  - Reunião de planejamento: Product Owner prioriza itens e equipe seleciona atividades para o Sprint
  - Cada Sprint se encerra com um incremento ao produto

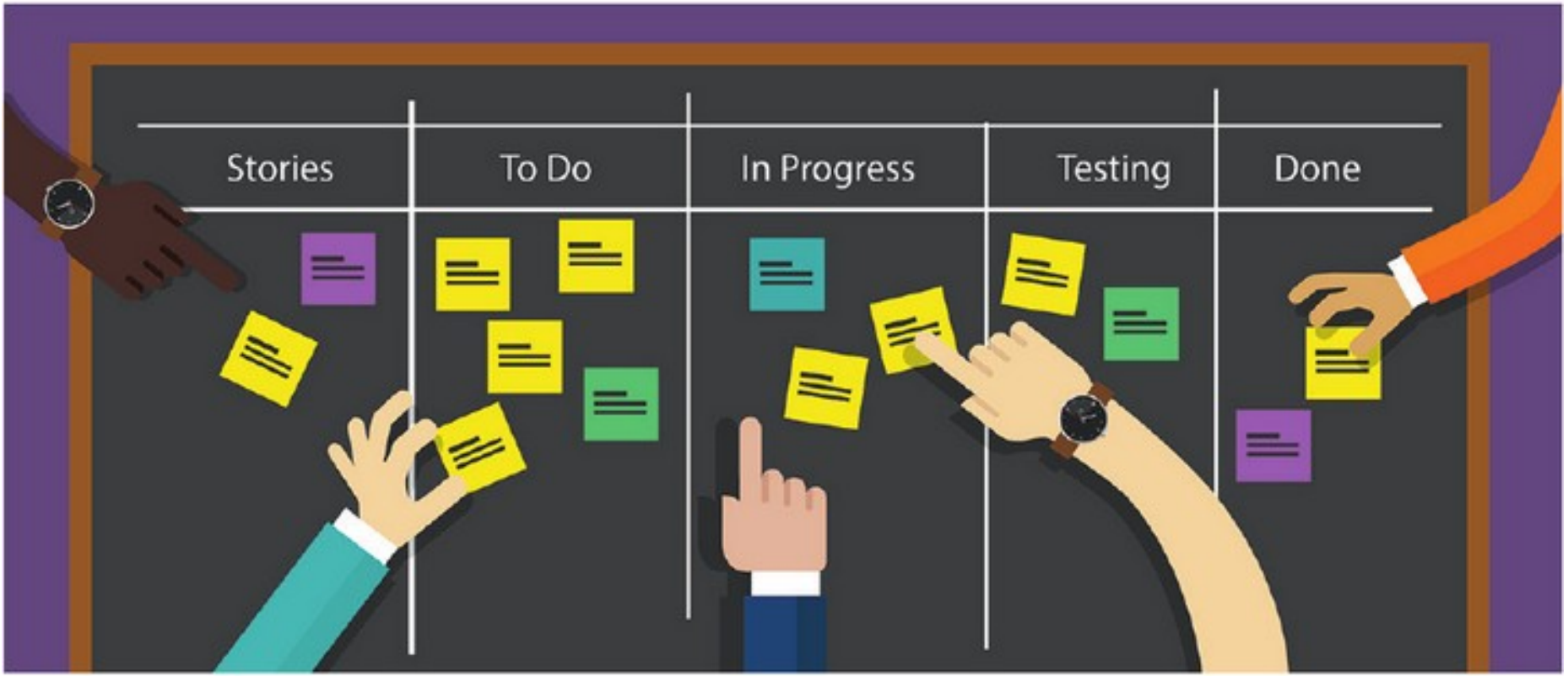
Scrum: Reuniões e Conclusão de Sprint

- Reuniões Scrum (Daily Meeting):
  - Reuniões breves, geralmente de 15 minutos
  - Realizadas diariamente (geralmente no início da manhã)
  - Três perguntas chaves para cada integrante:
    - O que você realizou desde a última reunião?
    - Quais obstáculos foram encontrados?
    - O que planeja realizar até a próxima reunião?
- Scrum Master:
  - Geralmente conduz as reuniões
  - Realiza avaliações das respostas, detectando problemas precocemente
- Término do Sprint (Pressman 2016):
  - Requisitos são concluídos e funcionamento é avaliado
  - Processo é melhorado para o Sprint sequencial

O Manifesto Ágil

- Base: todos os métodos de desenvolvimento Ágil são baseados no Manifesto Ágil
- Quatro valores fundamentais:
  - Primeiro: indivíduos e suas interações são mais importantes que processos e ferramentas
  - Segundo: software que funciona é mais importante que documentação vasta
- Mais valores:
  - Terceiro: a colaboração do cliente é mais importante que negociação de contratos
  - Quarto: responder às mudanças solicitadas é mais importante que seguir um plano

Quadro Scrum



O Quadro Scrum

- Ferramenta visual: quadro (board) montado para acompanhar as tarefas
  - Pode ser adaptado para a realidade de cada equipe
- Colunas de atividades:
  - Stories: descrição das necessidades do cliente (requisito explicado pelo usuário)
  - To Do (A Fazer): tarefas que deverão ser realizadas
- Colunas de status:
  - In Progress (Em Andamento): tarefas em andamento, mas não finalizadas
  - Testing: módulos que estão na fase de teste
  - Done (Pronto): relação do que já foi finalizado no sistema
  - O quadro reflete o fluxo de trabalho da equipe

Análise da Missão: Projeto Loja de Brinquedos Online

- Complexidade do projeto: cliente deseja site (e-commerce) e aplicativo
  - App com funcionalidades adicionais (aluguel apenas no app)
  - Provável necessidade de equipe subdividida para site e app
  - Considerar Android e iOS pode exigir equipe para cada SO
- Questões preliminares (antes da escolha do modelo):
  - O cliente tem pressa?
  - A equipe de desenvolvimento é grande o suficiente?
  - A equipe domina a tecnologia envolvida?
- Recomendação preliminar: alta probabilidade de um modelo baseado em processos ágeis
  - Participação do cliente é essencial para o sucesso
  - Divide o Software para entregas em partes menores
  - Cliente pode participar ativamente e ver resultados rapidamente

Conclusão: Proposta de Modelo e Próximos Passos

- Modelo sugerido: metodologia Scrum
  - Entregas feitas a cada sprint: cliente sabe o que e quando receberá
  - Requisitos mais importantes podem ser entregues primeiro
  - Permite que site e aplicativo comecem a operar rapidamente
  - Quadro Scrum ajuda a organizar cronograma e equipes
- Exemplo de quadro Scrum (para o site):
  - Backlog com tarefas: gerar interfaces (Wireframes), definir paleta de cores, definir estrutura do banco de dados
- Melhor processo de software (Pressman 2016):
  - Aquele próximo à equipe de desenvolvimento
  - Cenário ideal é propor um processo de software que se adapte às necessidades da equipe