Os **modelos incrementais de processo** são indicados quando os requisitos do software estão relativamente bem definidos, mas uma abordagem linear não é viável.

Nessas situações, é essencial fornecer rapidamente uma versão funcional do software e aprimorá-la continuamente. Esse método permite entregas progressivas, melhor adaptação a mudanças e maior envolvimento dos usuários.

Entre os principais modelos incrementais estão:

- **Modelo incremental**: desenvolvimento em etapas, com cada incremento adicionando novas funcionalidades até a conclusão do sistema.
- **Modelo RAD (Rapid Application Development)**: foca na entrega ágil e interativa, com ciclos curtos e intensa participação dos usuários para refinamento contínuo.

O **modelo incremental** combina elementos do modelo em cascata com a filosofia iterativa da prototipação, permitindo entregas parciais e operacionais a cada incremento.

Suas principais características incluem:

- Aplicação de sequências lineares de forma progressiva.
- Cada incremento adiciona novas funcionalidades ao software.
- Os primeiros incrementos são versões simplificadas do produto final.
- O primeiro incremento, chamado **núcleo do produto (core)**, contém as funções essenciais.

Um exemplo clássico é o desenvolvimento de um processador de texto, onde os incrementos adicionam funcionalidades como edição básica, formatação avançada, verificação gramatical e layout aprimorado. Esse ciclo se repete até que o software esteja completo.

O **modelo RAD (Rapid Application Development)** é uma adaptação do modelo em cascata que prioriza a alta velocidade no desenvolvimento por meio da modularização do sistema e reutilização de componentes.

Principais características:

- Indicado para projetos com requisitos bem definidos e objetivos restritos.
- Permite criar sistemas funcionais em períodos curtos (60 a 90 dias).
- Divide o desenvolvimento em equipes que trabalham em paralelo.
- Segue as fases: **especificação, planejamento, modelagem, construção e implantação**.

Vantagens e aplicações:

- Ideal para projetos com prazos rígidos.
- Funciona bem quando o sistema pode ser modularizado e entregue em partes.
- Adequado para equipes organizadas e comprometidas com entregas rápidas.

Limitações:

- Exige um grande número de desenvolvedores para dividir tarefas.
- Depende de forte colaboração entre clientes e equipe de desenvolvimento.
- Não é recomendado para projetos de alta complexidade técnica ou com tecnologias desconhecidas.

Desenvolvimento baseado em componentes

O **desenvolvimento baseado em componentes (CBD – Component-Based Development)**, também chamado de **Component-Based Software Engineering (CBSE)**, utiliza módulos de software reutilizáveis para acelerar a criação de sistemas.

Principais características:

- Baseado na reutilização de componentes prontos, disponíveis em bibliotecas gratuitas ou comerciais (**COTS – Commercial Off-The-Shelf**).
- Segue uma abordagem iterativa, combinando conceitos do modelo espiral.
- Integra componentes pré-existentes ao software em desenvolvimento.
- Utiliza conceitos da programação orientada a objetos, permitindo a criação de bibliotecas de classes reutilizáveis.

Vantagens:

- Redução significativa do tempo de desenvolvimento (até 70%).
- Diminuição dos custos do projeto (até 84%).
- Aumento da produtividade dos desenvolvedores.

Esse modelo é ideal para projetos que podem se beneficiar do **reuso de software**, garantindo maior eficiência e qualidade no desenvolvimento.

O **reuso de software** consiste na utilização de componentes já existentes para desenvolver novos sistemas, reduzindo custos e tempo de desenvolvimento. A decisão de reutilizar componentes deve considerar o investimento necessário e os benefícios obtidos em comparação ao desenvolvimento de soluções personalizadas.

Métodos e abordagens para reuso de software:

- **RiSE (Reuse in Software Engineering):** foca na redução de custos e no aumento da produtividade sem comprometer a qualidade do software.
- **Cruise (Component Reuse in Software Engineering):** livro online que mapeia diversos aspectos do reuso, incluindo componentes, reengenharia, ferramentas e métricas.
- **RAD (Rapid Application Development):** modelo incremental que acelera o desenvolvimento ao reutilizar componentes já existentes ou criar novos reutilizáveis.

O reuso de software é uma estratégia essencial para otimizar projetos, garantindo eficiência e qualidade no desenvolvimento.