



# Front-end Frameworks

Prof. Msc. Igor Revoredo

## Unidade 1

*Introdução à linha de produção  
de software*



**UNINASSAU**

# Objetivos

- *Conceituar linha de produção de software.*
- *Diferenciar os frameworks das linhas de produção de software.*
- *Exemplificar a arquitetura de uma linha de produção de software.*

# Contexto

A abordagem de reúso de software era considerada incomum até por volta do ano 2000.

A crescente demanda por entrega de software com (1) prazos cada vez mais pequenos, (2) a redução de custos de produção e de manutenção e (3) a exigência de alta qualidade de software **contribuíram para que o reúso sistemático de software se tornasse uma prática comum.**

Portanto, ao longo dos anos **várias abordagens de reutilização de software** foram propostas, incluindo: paradigmas de orientação a objetos (polimorfismo, encapsulamento e herança), padrões de projeto e de arquitetura, uso de componentes, frameworks de aplicações, entre outras.

Entretanto, como essas abordagens tradicionais não foram totalmente satisfatórias, surgiu a necessidade de propor uma nova abordagem, conhecida como **linha de produtos de software** (software product line — SPL).

# **Linha de produtos de software**

## **(software product line — SPL)**

- A ideia básica do desenvolvimento baseando em SPL é **suportar o reúso de software em grande escala.**
- Para isso, **conhecimentos da área de negócios** são recrutados para separar as partes semelhantes de um conjunto de produtos relacionados que atendem a um determinado segmento de mercado.
- Desse modo, torna-se possível desenvolver software de forma mais eficiente em termos de **custo, qualidade e tempo.**

# **Linha de produtos de software**

## **(software product line — SPL)**

- trata-se da reutilização de algo já construído, com o objetivo de agilizar o desenvolvimento de software e também reduzir custos.
- É importante salientar que o reúso não diz respeito apenas à reutilização dos componentes, pois também engloba todo o planejamento para que isso ocorra da forma mais eficiente possível.

# **Linha de produtos de software**

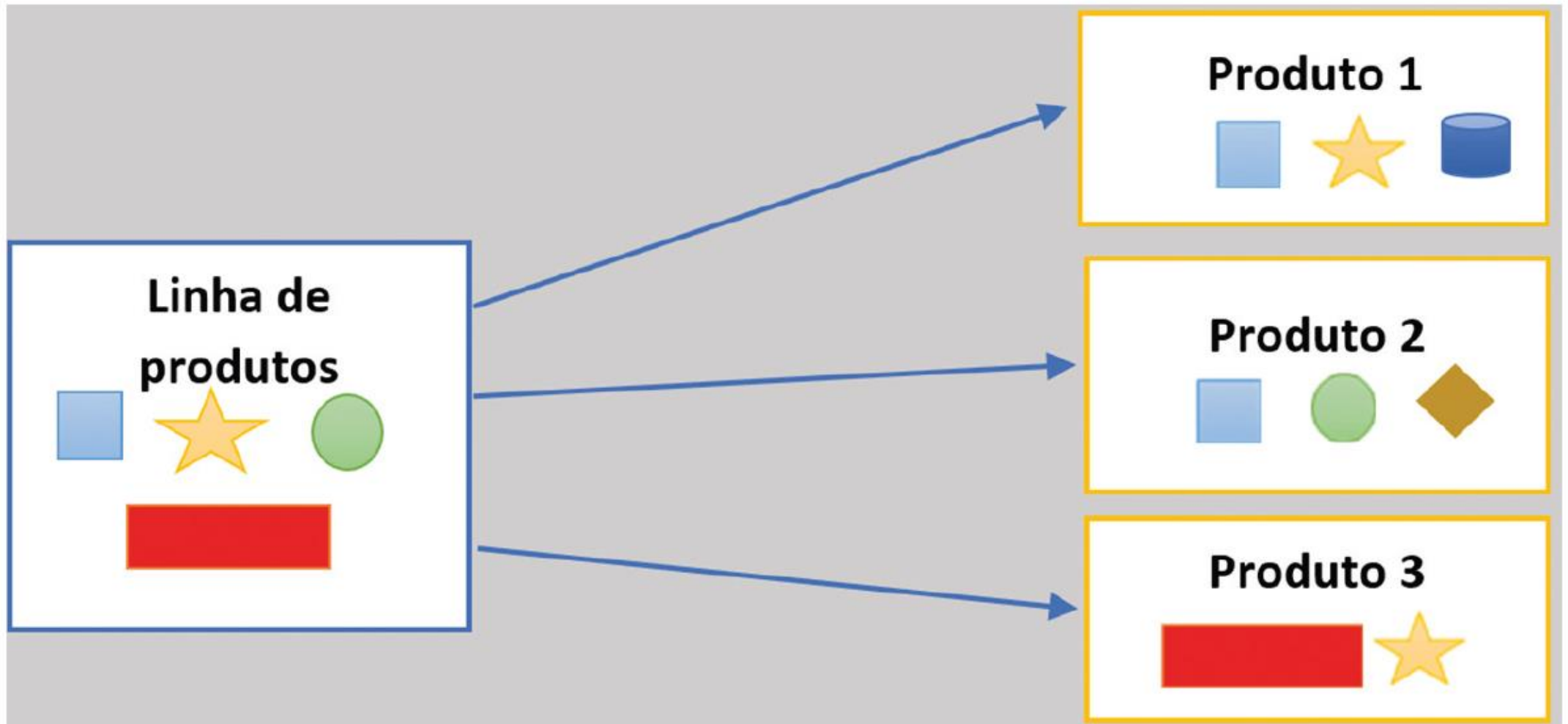
## **(software product line — SPL)**

existem diversas abordagens que auxiliam o reuso de software, incluindo:

- bibliotecas,
- padrões de projeto e de arquitetura,
- frameworks de aplicação,
- sistemas de sistemas, integração de sistemas de aplicação (COTS),
- sistemas Enterprise Resource Planning (ERP),
- sistemas de aplicação configuráveis,
- empacotamento de sistemas legados,
- engenharia baseada em componentes,
- engenharia dirigida por modelos,
- sistemas orientados a serviços, engenharia de software orientada a aspectos,
- gerador de programas e SPL.

# Linha de produtos de software

(software product line — SPL)



Exemplificação genérica de uma SPL.



# **Linha de produtos de software**

## **(software product line — SPL)**

- De acordo com Sommerville (2019, p. 413) a SPL é definida como “[...] um conjunto de aplicações com uma arquitetura comum e componentes compartilhados, em que cada aplicação se especializa em refletir requisitos específicos do cliente”.
- Em suma, pode-se definir SPL como uma técnica para construir softwares baseados em componentes que podem ser reutilizados para construir outros softwares sem muito esforço.



**ESSA É A  
PARTE MAIS  
IMPORTANTE  
DO REACTJS**



# **Linha de produtos de software**

## **(software product line — SPL) VANTAGENS**

existem inúmeros benefícios para o uso da SPL, com destaque para:

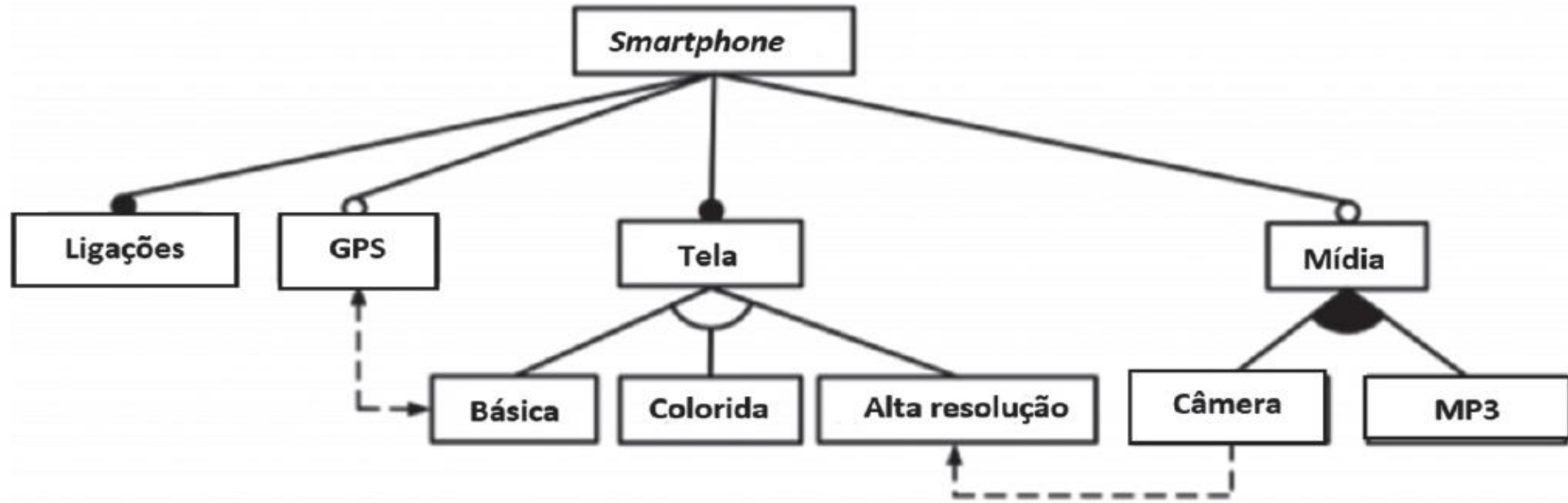
- produção de software em larga escala,
- redução do tempo de lançamento do produto de software no mercado,
- melhoria na qualidade do produto,
- aumento da satisfação do cliente,
- capacidade de personalizar ou configurar rapidamente o produto de software e
- até a capacidade de amparar um crescimento sem precedentes.







# **Linha de produtos de software**

## **(software product line — SPL) DESVANTAGENS**

- No entanto, a SPL também apresenta algumas desvantagens, pois não se encaixa nos planos de desenvolvimento de todas as organizações, sobretudo naquelas inseridas em mercados que passam por mudanças frequentes e cujos produtos estão cada vez mais diversificados.
- Há também os produtos que se inserem em domínios com futuros incertos e, portanto, representam um grande risco para o sucesso de uma organização de SPL (KAKOLA; DUEÑAS, 2006).

# Abstração



	Obrigatória		Alternativa (XOR)		Requer (Requires)
	Opcional		Alternativa (OR)		Exclui (Excludes)

# Diferenças entre frameworks e SPL

- Os primeiros entusiastas do desenvolvimento orientado a objetos sinalizaram que seu uso poderia trazer muitos benefícios, visto que poderiam reusar os objetos em diferentes softwares.
- Entretanto, com o passar dos anos essa recomendação se mostrou falsa, pois os objetos são muitas vezes granulares e muito particularizados para um determinado software
- e, normalmente, acaba-se consumindo mais tempo para entendê-los e adaptá-los para reaproveitamento do que para implementá-los do zero.



# Diferenças entre frameworks e SPL

- A experiência com a abordagem de reuso de software comprovou que é mais acertado utilizar essa abordagem em um processo de desenvolvimento orientado a objetos por meio de abstrações mais genéricas, como *frameworks*.
- um framework é definido como um conjunto integrado de artefatos de software (como classes, objetos e componentes) que colaboram para proporcionar uma arquitetura reusável para uma família de aplicações relacionadas.

# Diferenças entre frameworks e SPL

- A experiência com a abordagem de reúso de software comprovou que é mais acertado utilizar essa abordagem em um processo de desenvolvimento orientado a objetos por meio de abstrações mais genéricas, como *frameworks*.
- um framework é definido como um conjunto integrado de artefatos de software (como classes, objetos e componentes) que colaboram para proporcionar uma arquitetura reusável para uma família de aplicações relacionadas.



# Abstração



**Projetos simples podem ser realizados por uma única pessoa**

Pouca  
modelagem

Ferramentas  
simples

Processo  
simples

Pouco projeto

Pouca  
especialização  
para construir

# Abstração



**Projetos complexos/maiores exigem  
arquitetura**

**Mais  
modelagem**

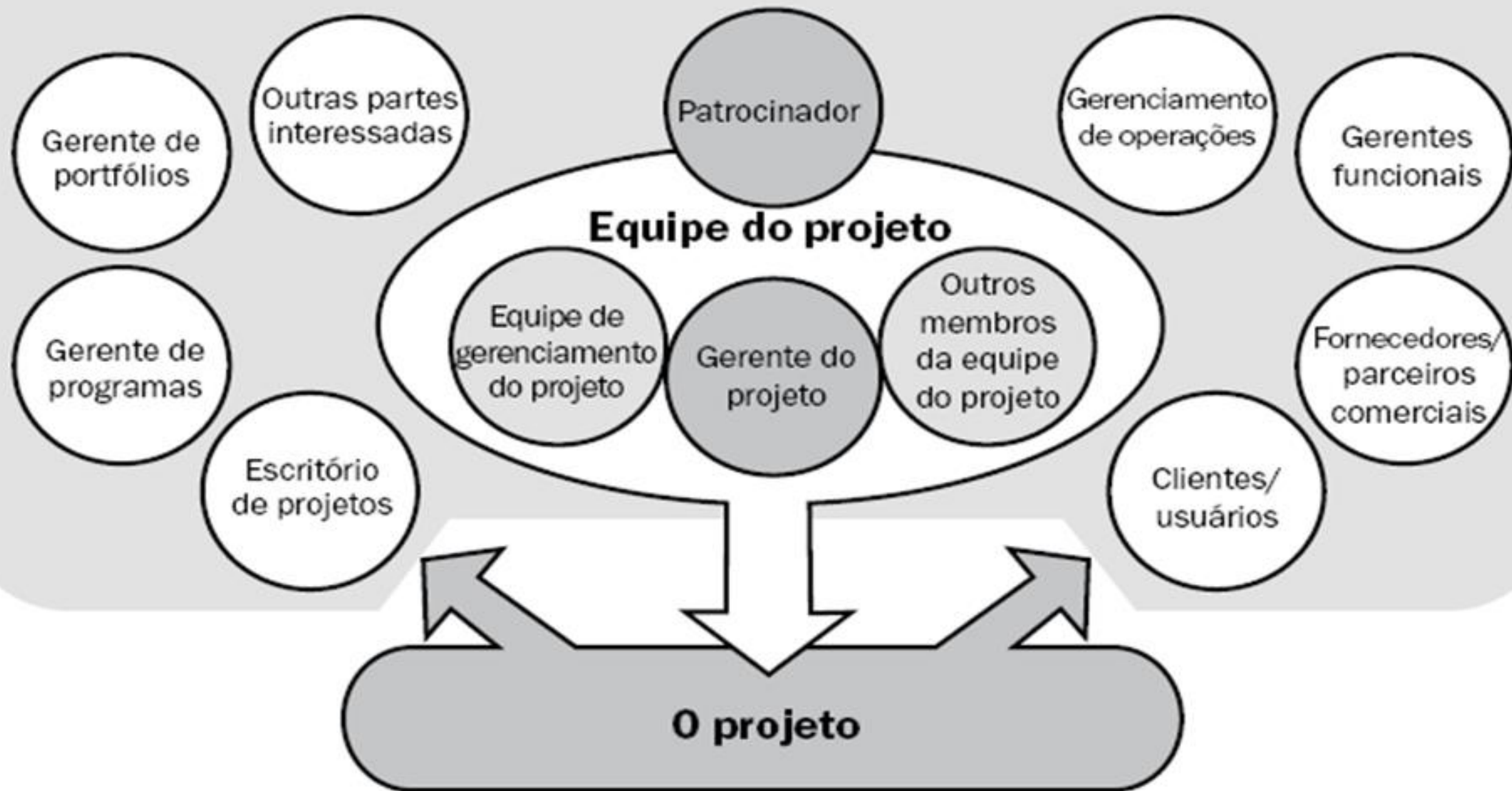
**Ferramentas  
mais  
poderosas**

**Processo mais  
bem definidos**

**Mais projeto**

**Alta  
especialização  
para construir**

## Partes interessadas no projeto







Como o cliente explicou...



Como o líder de projeto entendeu...



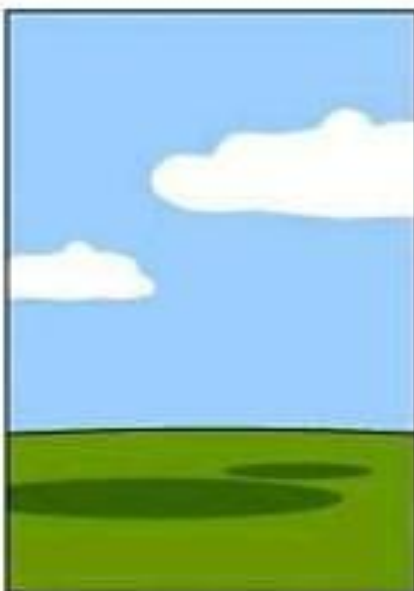
Como o analista projetou...



Como o programador construiu...



Como o Consultor de Negócios descreveu...



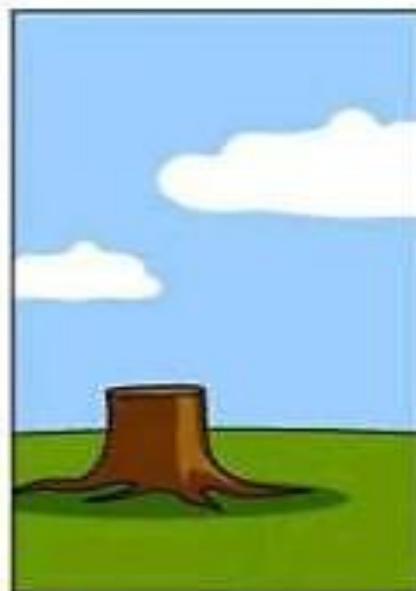
Como o projeto foi documentado...



Que funcionalidades foram instaladas...



Como o cliente foi cobrado...



Como foi mantido...



O que o cliente realmente queria...