Engenharia de Software

Ma. Vanessa Matias Leite

Unidades de software reutilizadas

- **Reuso de sistemas:** Sistemas completos, compostos por uma série de programas de aplicação, podem ser reutilizados como parte de um sistema de sistemas.
- **Reuso de aplicações:** Uma aplicação pode ser incorporada sem alterações em outros sistemas ou configurada para atender a diferentes clientes.
- **Reuso de componentes:** Componentes de uma aplicação, desde subsistemas até objetos únicos, podem ser reutilizados.
- **Reuso de objetos e funções:** Componentes de software que implementam uma única função, como uma função matemática, ou uma classe de objeto, podem ser reutilizados.

Benefício	Explicação
Desenvolvimento acelerado	Lançar um sistema no mercado o mais brevemente possível costuma ser mais importante do que os custos totais do desenvolvimento. O reúso de software pode acelerar a produção do sistema, pois tanto o tempo de desenvolvimento quanto o de validação podem ser reduzidos.
Uso eficaz de especialistas	Em vez de refazer o mesmo trabalho repetidamente, os especialistas de aplicação podem desenvolver um software reusável que encapsule o seu conhecimento.
Maior dependabilidade	O software reusado, que foi testado e aprovado em sistemas em funcionamento, deve ser mais confiável do que o novo software. Seus defeitos de projeto e implementação devem ter sido descobertos e corrigidos.
Custos de desenvolvimento mais baixos	Os custos de desenvolvimento são proporcionais ao tamanho do software que está sendo desenvolvido. Reusar o software significa que menos linhas de código têm de ser escritas.
Menos risco para o processo	O custo do software existente já é conhecido, enquanto os custos de desenvolvimento são sempre uma incógnita. Esse fator é importante para o gerenciamento do projeto porque reduz a margem de erro na estimativa de custo do projeto. Isso vale especialmente quando grandes componentes de software, como os subsistemas, são reusados.
Conformidade com os padrões	Alguns padrões, como os de interface com o usuário, podem ser implementados como um conjunto de componentes reusáveis. Por exemplo, se os menus em uma interface com o usuário forem implementados usando componentes reusáveis, todas as aplicações apresentam os mesmos formatos de menu para os usuários. O uso de um padrão de interface com o usuário melhora a dependabilidade, porque os usuários cometem menos erros quando é apresentada a eles uma interface conhecida.

Fonte: Sommerville (2018).

Problema	Explicação
Criar, manter e usar uma biblioteca de componentes	Povoar uma biblioteca de componentes reusáveis e garantir que os desenvolvedores de software possam usar essa biblioteca pode custar caro. Os processos de desenvolvimento têm de ser adaptados para garantir que a biblioteca seja utilizada.
Encontrar, entender e adaptar componentes reusáveis	Os componentes de software têm de ser descobertos em uma biblioteca, compreendidos e, às vezes, adaptados para trabalhar em um novo ambiente. Os engenheiros devem estar razoavelmente confiantes de que encontrarão um componente na biblioteca antes de incluírem uma busca por componentes como parte de seu processo de desenvolvimento normal.
Maiores custos de manutenção	Se o código-fonte de um sistema de software ou componente reusado não estiver disponível, então os custos de manutenção podem ser mais altos porque os elementos reusados do sistema podem se tornar incompatíveis com as mudanças feitas no sistema.
Falta de suporte da ferramenta	Algumas ferramentas de software não fornecem suporte ao desenvolvimento com reúso. Pode ser difícil ou impossível integrar essas ferramentas com um sistema de biblioteca de componentes. O processo de software empregado por essas ferramentas pode não levar em conta o reúso. É mais provável que isso aconteça com as ferramentas que apoiam a engenharia de sistemas embarcados do que com as ferramentas de desenvolvimento orientado a objetos.
Síndrome do 'não inventado aqui'	Alguns engenheiros de software preferem reescrever os componentes porque acreditam que podem aperfeiçoá-los. Isso tem a ver em parte com a confiança e em parte com o fato de que escrever o software original é encarado como algo mais desafiador do que reusar o software de outras pessoas.

Fonte: Sommerville (2018).

Panorama de Reuso

Fonte: Sommerville (2018).

Abordagens que apoiam o reuso

Abordagem	Descrição
Frameworks de aplicação	Coleções de classes abstratas e concretas são adaptadas e estendidas para criar sistemas de aplicação.
Integração de sistemas de aplicação	Dois ou mais sistemas de aplicação são integrados para proporcionar mais funcionalidade.
Padrões de arquitetura	Padrões de arquiteturas de software que apoiam tipos comuns de sistemas de aplicação são utilizados como base de aplicações.
Desenvolvimento de software orientado a aspectos	Componentes compartilhados são 'entrelaçados' em uma aplicação em diferentes lugares quando o programa é compilado.
Engenharia de software baseada em componentes	Sistemas são desenvolvidos integrando componentes (coleções de objetos) que se adaptam aos padrões de modelo de componentes.
Sistemas de aplicação configuráveis	Sistemas específicos de domínio são projetados para que possam ser configurados para as necessidades de clientes de sistemas específicos.
Padrões de projeto	Abstrações genéricas que ocorrem entre aplicações são representadas como padrões de projeto que exibem objetos e interações abstratos e concretos.

Fonte: Sommerville (2018).

Planejar o Reuso

Cronograma de desenvolvimento do software

Tempo de vida previsto para o software

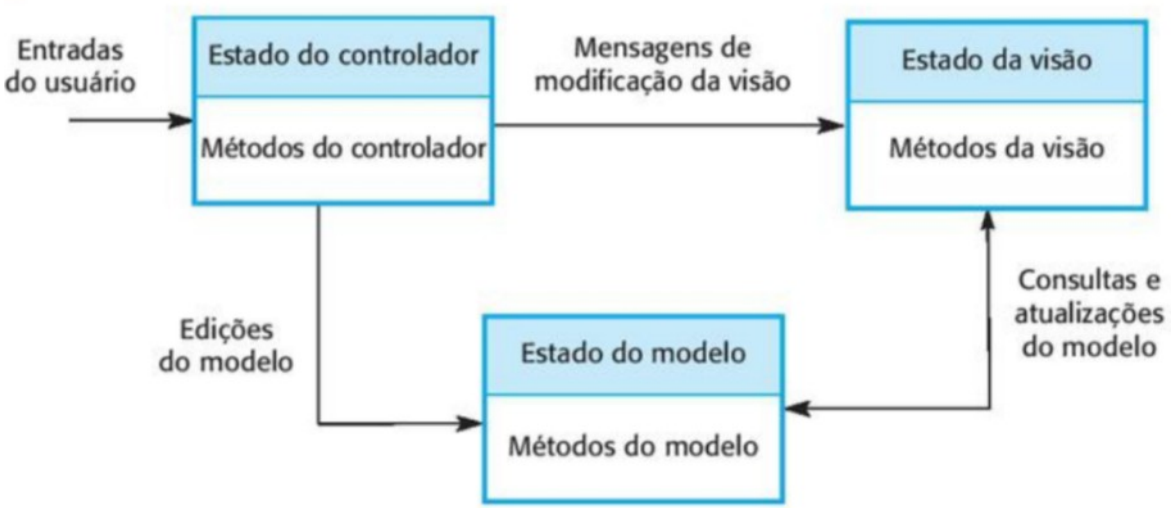
Habilidades da equipe de desenvolvimento

Criticidade do software

Domínio de aplicação

Plataforma na qual o sistema será executado

Padrão MVC



Fonte: Sommerville (2018)