# Engenharia de Software III

## Aula 2
## Arquitetura de Software

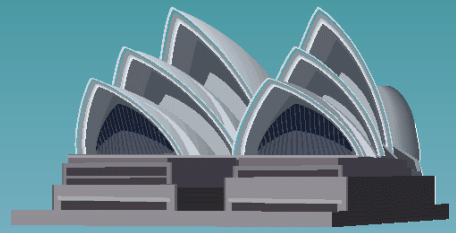
l.bertholdo@ifsp.edu.br

# Conteúdo

- Arquitetura de Software
- Projeto de Arquitetura
- Decisões de Projeto de Arquitetura
- Visões da Arquitetura
- Documentação de Arquitetura
- Agilidade e Arquitetura

*l.bertholdo@ifsp.edu.br*

# Arquitetura de Software



- Quando pensamos na arquitetura de um edifício, logo vem à mente a forma de sua estrutura física e suas características estéticas.

- Porém, arquitetura também descreve como os vários componentes do edifício são integrados para formar um todo coeso, e como o edifício se ajusta ao seu ambiente adjacente. Detalha também o projeto de iluminação, o tipo de piso, o posicionamento de painéis etc.

- Além disso, arquitetura também se refere a **centenas de decisões**, tanto grandes quanto pequenas.

*Algumas das decisões da arquitetura são tomadas logo no início do projeto e podem ter um impacto profundo sobre todas as ações subsequentes. Outras são postergadas ao máximo, para eliminar restrições que levariam a uma implementação inadequada do estilo arquitetônico.*

*l.bertholdo@ifsp.edu.br*

# Arquitetura de Software

- A **arquitetura de software** se assemelha à arquitetura da construção civil em muitos aspectos:

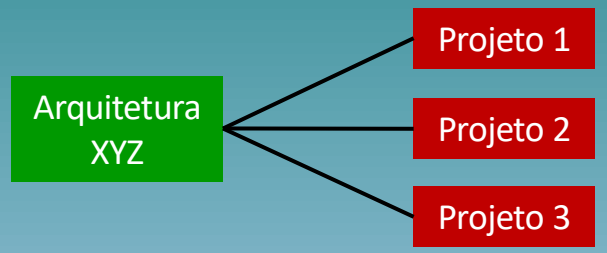
> *Ela representa a **estrutura geral** de um sistema, a qual abrange os **componentes de software**, as propriedades **externamente visíveis** desses componentes e as **relações** entre eles.*

- Um **componente de software** pode ser algo tão simples quanto um módulo de programa ou uma classe orientada a objetos, mas também pode ser ampliado para abranger bancos de dados e *middleware* que permitam a configuração de uma rede cliente/servidor.

- As **propriedades dos componentes** são necessárias para entender como eles interagem entre si. Porém, no nível da arquitetura, as **propriedades internas** não são especificadas (por exemplo, detalhes de um método).

*l.bertholdo@ifsp.edu.br*

# Arquitetura de Software




- Embora os termos “arquitetura” e “projeto” possam se confundir, há uma diferença marcante entre eles. **Projeto** é uma instância de uma **arquitetura**, assim como um **objeto** é uma instância de uma **classe**.

*Supondo uma arquitetura cliente/servidor, podemos projetar um sistema de várias formas por meio dessa arquitetura, por exemplo, usando a plataforma Java ou a plataforma .NET. Ou seja, vários projetos podem ser criados com base em uma arquitetura.*

- Embora cada projeto tenha suas particularidades, os **elementos** e **estruturas** que definem a arquitetura são a raiz de qualquer projeto. Por isso, todo **projeto** se inicia necessariamente com base em alguma **arquitetura**.

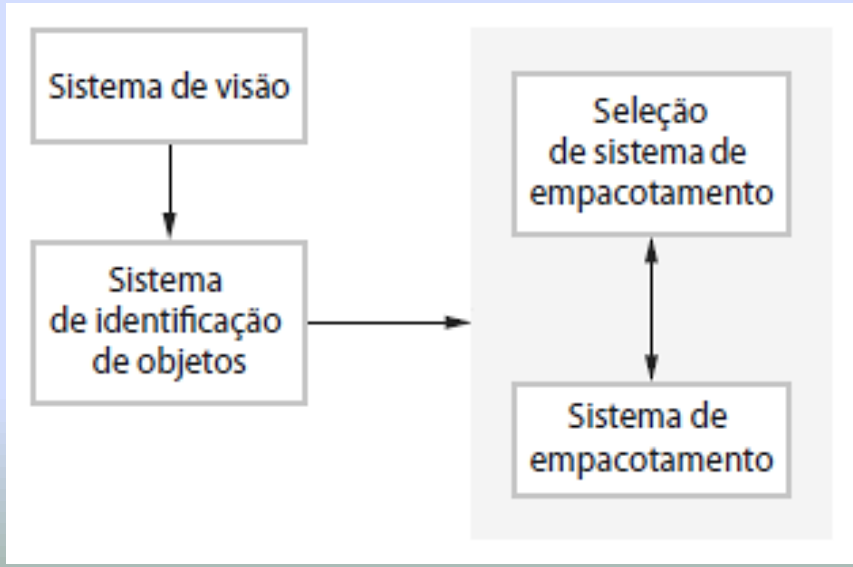
# Arquitetura de Software

- Por que a arquitetura de software é importante?
  - ✓ Fornece uma representação de alto nível que facilita a comunicação entre os diferentes envolvidos.
  - ✓ Permite analisar a efetividade do projeto com relação ao atendimento dos requisitos declarados.
  - ✓ Afeta vários requisitos não funcionais como: desempenho, robustez, confiabilidade, manutenibilidade, entre outros.
  - ✓ Explicita, desde o início, as decisões de projeto que terão um profundo impacto no trabalho de engenharia de software que se segue.
  - ✓ Propicia um momento para avaliar alternativas de arquitetura em um estágio em que fazer mudanças de projeto ainda é razoavelmente fácil.
  - ✓ Pode contribuir para o reúso de software, já que a arquitetura geralmente é a mesma para sistemas com requisitos semelhantes.

*l.bertholdo@ifsp.edu.br*

# Projeto de Arquitetura


- O **projeto de arquitetura** é o primeiro estágio no processo de **projeto de software**. Ele define como um sistema deve ser organizado e como será sua estrutura geral.
- A **arquitetura** é o elo crítico entre o **projeto** e a **engenharia de requisitos**, pois identifica os principais componentes estruturais do sistema, e os relacionamentos entre eles, com base nos requisitos.



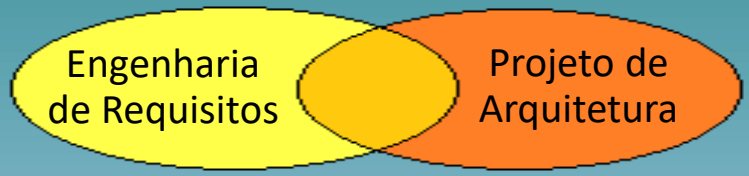

***Sistema Robotizado de Empacotamento***

*Este modelo abstrato de arquitetura mostra os componentes que precisam ser desenvolvidos e os relacionamentos entre eles.*

*O sistema usa um componente para visualizar e selecionar objetos em uma esteira, outro para identificar o tipo de objeto, e um terceiro que engloba subcomponentes para selecionar o tipo correto de empacotamento e realizar o empacotamento.*

l.bertholdo@ifsp.edu.br

# Projeto de Arquitetura

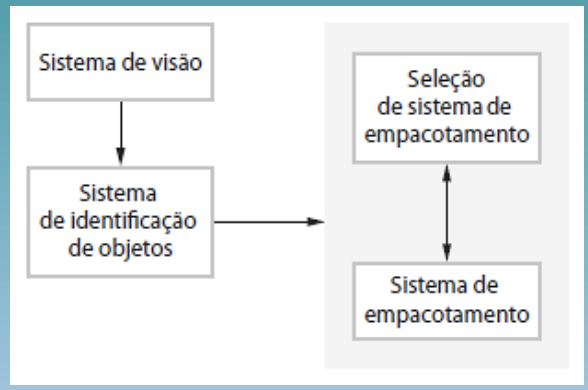



- Na prática, existe uma considerável **sobreposição** entre os processos de engenharia de requisitos e de projeto de arquitetura.
- Normalmente, uma especificação de sistema tem como foco os requisitos de software. No entanto, a decomposição de arquitetura é geralmente necessária para estruturar e organizar a especificação.
- Por isso, como parte do processo de engenharia de requisitos, muitas vezes é preciso propor uma **arquitetura abstrata de alto nível**, em que seja possível associar grupos de funções ou recursos aos **módulos do sistema** e, eventualmente, sistemas externos.

*A decomposição da arquitetura pode ser usada inclusive para discutir com os envolvidos os requisitos e recursos do sistema.*

*l.bertholdo@ifsp.edu.br*

# Projeto de Arquitetura




- Inicialmente, a arquitetura pode ser modelada por meio de **diagramas de blocos simples**, onde cada caixa representa um componente.
- Caixas dentro de caixas indicam que o componente foi decomposto em **subcomponentes**. E as setas indicam a direção em que os dados e/ou mensagens são passados de um componente a outro.
- Estes diagramas apresentam uma imagem de alto nível da estrutura do sistema, de forma que os diferentes envolvidos no processo de desenvolvimento possam compreendê-la facilmente.

*Apesar de seu amplo uso, esses diagramas são **representações pobres de arquitetura**, pois não mostram o tipo de relacionamento entre os componentes do sistema, nem as propriedades externas dos componentes. Por isso, se a arquitetura de um sistema precisa ser bem documentada, é melhor usar uma **notação mais detalhada** para descrever a arquitetura.*

# Decisões de Projeto de Arquitetura

- Durante o projeto de arquitetura, os arquitetos de software precisam tomar uma série de **decisões estruturais** que afetam profundamente o sistema e seu processo de desenvolvimento. Entre as **questões essenciais** que precisam ser consideradas estão:

1. *Existe uma arquitetura genérica de alguma outra aplicação que pode servir de modelo para o sistema que está sendo projetado?*
2. *Como o sistema será distribuído entre os servidores?*
3. *Que padrões ou estilos de arquitetura podem ser usados?*
4. *Como os componentes do sistema serão decompostos em subcomponentes?*
5. *Que estratégia será usada para controlar o funcionamento dos componentes do sistema?*
6. *Qual a melhor organização de arquitetura para satisfazer os requisitos não funcionais do sistema?*
7. *Como o projeto de arquitetura será avaliado?*
8. *Como a arquitetura do sistema deve ser documentada?*

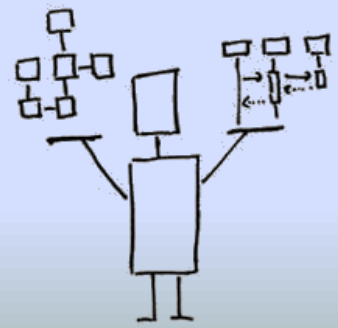
# Decisões de Projeto de Arquitetura

- A arquitetura de um sistema de software pode se basear em um determinado **padrão ou estilo de arquitetura**.

- **Padrão de arquitetura** é uma descrição da organização de um sistema, que permitirá o cumprimento de seus requisitos. Alguns exemplos são a estrutura cliente-servidor e a arquitetura em camadas.

- Cada padrão de arquitetura captura a essência de um modelo de arquitetura particular, o qual vem historicamente sendo usado em diferentes sistemas de software.

*Ao tomar decisões sobre a arquitetura de um sistema, é preciso conhecer os padrões comuns, bem como saber em que tipo de projeto eles podem ser usados e quais são seus pontos fortes e fracos.*



*l.bertholdo@ifsp.edu.br*

# Decisões de Projeto de Arquitetura

- Os **requisitos não funcionais** e a **arquitetura do software** têm uma estreita relação. Por isso, o estilo e a estrutura da arquitetura escolhida para um sistema devem depender deste tipo de requisito.
- Por exemplo, supondo que cada requisito a seguir seja crítico:

***Desempenho:*** *A arquitetura pode ser projetada para colocar as operações críticas dentro de um pequeno número de componentes, implantados no mesmo servidor, em vez de distribuídos pela rede.*

***Proteção:*** *Deve ser usada uma arquitetura em camadas, com os ativos (informações) mais críticos protegidos nas camadas mais internas, as quais teriam alto nível de validação.*

***Disponibilidade:*** *A arquitetura deve ser projetada para incluir componentes redundantes, de modo que seja possível substituir e atualizar componentes sem parar o sistema.*

***Segurança:*** *As operações relacionadas à segurança devem estar em um único (ou poucos) componente(s). Isso reduz os custos e os problemas de validação, e permite definir um comportamento adequado em caso de falhas, como desligar o sistema de forma segura.*

***Manutenibilidade:*** *A arquitetura deve ser projetada a partir de componentes autocontidos, que podem ser facilmente alterados. Os produtores de dados devem ser separados dos consumidores, e as estruturas de dados compartilhados evitadas.*

*l.bertholdo@ifsp.edu.br*

# Decisões de Projeto de Arquitetura

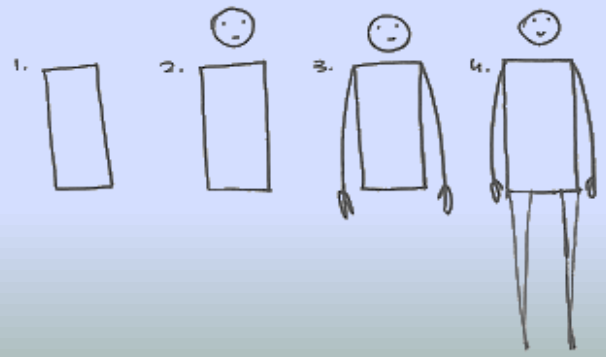
- Obviamente, há um conflito entre algumas dessas arquiteturas. Por exemplo, o uso de componentes grandes melhora o **desempenho**. Por outro lado usar muitos componentes pequenos contribui para a **manutenibilidade**.

- Se ambos os requisitos não funcionais forem importantes no sistema, será preciso achar um meio-termo. Uma solução possível seria usar **estilos de arquitetura distintos para diferentes partes do sistema**.

*Avaliar um projeto de arquitetura é difícil, pois o verdadeiro teste de uma arquitetura é* ***quão bem o sistema satisfaz aos requisitos funcionais e não funcionais*** *quando ele está em uso. No entanto, é possível fazer uma avaliação, analisando o quanto o projeto está aderente às arquiteturas de referência ou padrões genéricos de arquitetura.*

# Visões da Arquitetura

- É impossível representar todas as informações relevantes sobre a arquitetura de um sistema em um **único modelo**, pois cada modelo mostra apenas **uma perspectiva do sistema**.
- Um modelo de arquitetura pode mostrar como um sistema é decomposto em módulos, como seus processos interagem ou como seus componentes são distribuídos em rede.
- Cada uma dessas informações é útil em momentos diferentes do processo de desenvolvimento.
- Portanto, o projeto e sua documentação precisam apresentar **múltiplas visões da arquitetura de software**.




*l.bertholdo@ifsp.edu.br*

# Visões da Arquitetura

- Embora não exista um consenso sobre quais visões de arquitetura são necessárias, existem quatro delas que são consideradas fundamentais:

***Visão lógica:*** *mostra abstrações básicas como pacotes e classes, permitindo relacionar os requisitos de sistema com as entidades representadas por estas classes.*

***Visão de processo:*** *mostra como os processos do sistema interagem em tempo de execução, auxiliando na análise das características não funcionais do sistema.*

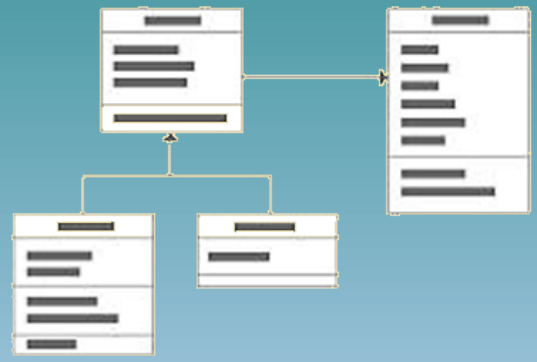
***Visão de desenvolvimento:*** *mostra como o software é decomposto em componentes para o desenvolvimento, sendo útil para gerentes de projeto e programadores.*

***Visão física:*** *mostra o hardware e como os componentes de software são distribuídos entre os processadores, informações importantes para a equipe de implantação.*

*Além dessas quatro visões, alguns autores citam a noção de* ***visão conceitual****, algo similar ao diagrama mostrado para o Sistema Robotizado de Empacotamento. Outros mencionam* ***visões comportamentais****, baseadas em casos de uso ou cenários. Ambas são usadas para* ***comunicar a essência de um sistema aos diferentes stakeholders****.*

l.bertholdo@ifsp.edu.br

# Visões da Arquitetura



- As visões usam **representações** para descrever a arquitetura do sistema, entre elas a **UML**.
- Embora a UML seja muitas vezes utilizada, no estágio do projeto de arquitetura, geralmente é preciso descrever os sistemas em um **nível maior de abstração**. Classes de objetos estão muito próximas da implementação para serem úteis na descrição de arquitetura.
- Por isso, muitos arquitetos preferem usar notações informais, rápidas de escrever e que são facilmente desenhadas em um quadro branco, deixando a UML apenas para documentar detalhes de arquitetura.

*Adeptos dos métodos ágeis alegam que a documentação detalhada de projeto é pouco usada, e desenvolvê-la é um desperdício de tempo e dinheiro. Exceto quando se está desenvolvendo um sistema crítico, em que uma análise detalhada é imprescindível, o ideal é desenvolver visões que ajudem na* ***comunicação do projeto****, sem se preocupar se o documento de arquitetura contém todos os detalhes.*

*l.bertholdo@ifsp.edu.br*

# Documentação de Arquitetura


- O propósito básico do documento de arquitetura é **registrar e comunicar as decisões** que foram tomadas para a escolha de uma determinada arquitetura.
- Esse documento oferece uma visão de alto nível sobre **questões técnicas** que servirão de base para o desenvolvimento do sistema, como os **componentes de software** que serão desenvolvidos para cada funcionalidade e as **tecnologias** que serão empregadas.
- Um documento de arquitetura deve descrever pelo menos:

  - *Os principais componentes do software e as suas interações;*
  - *Os estilos arquitetônicos a serem utilizados;*
  - *As plataformas de hardware e software para as quais o sistema será concebido e implementado;*
  - *Como a arquitetura contemplará os requisitos não funcionais.*

*l.bertholdo@ifsp.edu.br*

# Documentação de Arquitetura

- Estrutura genérica de um Documento de Arquitetura de Software

| Seção | Descrição |
|---|---|
| Introdução | Fornece uma visão geral do documento, descrevendo de forma sucinta seu objetivo e seu escopo. |
| Definições, acrônimos e abreviaturas | Contém as definições de todos os termos, acrônimos e abreviaturas necessários para interpretar corretamente o documento. |
| Representação arquitetural | Apresenta a arquitetura a ser utilizada na aplicação, mencionando as visões necessárias no projeto: conceitual/comportamental, lógica, de processo, de desenvolvimento e/ou física. Também informa como essas visões serão representadas no documento (normalmente por meio de diagramas da UML). |
| Objetivos e restrições da arquitetura | Lista os objetivos e requisitos de software que podem ter algum impacto sobre a arquitetura, como segurança, portabilidade, reúso etc. Também apresenta as premissas (restrições) definidas para implementação destes objetivos e requisitos. Tais restrições podem estar relacionadas, por exemplo, à *design* e implementação, ferramentas de desenvolvimento, códigos legados etc. |

# Documentação de Arquitetura

- Estrutura genérica de um Documento de Arquitetura de Software (cont.)

| Seção | Descrição |
|---|---|
| Visão conceitual / comportamental (de casos de uso) | Apresenta uma **visão conceitual** abstrata, baseada em diagramas de blocos, da arquitetura do sistema, ou uma **visão comportamental** dos casos de uso mais significativos, de modo a facilitar o entendimento dos cenários mais complexos que impactam a arquitetura. |
| Visão lógica | Descreve a divisão do sistema em módulos e pacotes. Para cada pacote, mostra suas respectivas classes, mencionando as principais responsabilidades, relacionamentos, operações e atributos destas classes. Normalmente, essa visão é ilustrada por meio de **diagramas de pacotes e de classes**. |
| Visão de processo | Apresenta a forma como os processos do sistema se comunicam em tempo de execução. Para isso, a descrição dessa visão costuma ser apoiada por **diagramas de sequência** ou **de comunicação**. |
| Visão de desenvolvimento (ou implementação) | Descreve a forma de implementação adotada para a solução descrita na visão lógica. São identificados os principais componentes, padrões de projeto, *frameworks*, ferramentas de desenvolvimento e o relacionamento entre esses elementos. Geralmente, essa visão faz uso de **diagramas de componentes**. |
| Visão física (ou de implantação) | Apresenta a configuração da rede física (hardware) na qual o software será implantado e executado, por meio de um **digrama de implantação** do sistema. |

# Documentação de Arquitetura

- Estrutura genérica de um Documento de Arquitetura de Software (cont.)

| Seção | Descrição |
|---|---|
| Capacidade e desempenho | Define as premissas (restrições) relacionadas à dimensionamento e performance. Por exemplo, volume de transações simultâneas que o sistema deve permitir, ou tempo máximo de resposta que o sistema deve levar para realizar uma determinada operação. |
| Diretrizes para especificação do projeto | Apresenta as diretivas que foram usadas como base para especificar o projeto de arquitetura, entre elas: notação usada para representar os componentes do projeto, tipos de diagramas usados, regras para nomenclatura dos componentes, nomenclatura dos diretórios de código-fonte e outros arquivos de implementação, além de outras diretrizes que forem pertinentes ao projeto. |
| Referências | Lista os documentos usados como insumo para a elaboração do documento de arquitetura (documentos de visão, especificações de requisitos etc). Cada documento deve ser identificado pelo título, número de identificação e versão. |
| Histórico de alterações do documento | Contém uma tabela com colunas para: (1) data de criação ou alteração do documento, (2) versão do documento, (3) nome do criador do documento (ou do autor da alteração) e (4) descrição resumida das alterações realizadas. |

# Agilidade e Arquitetura

- Embora muitos desenvolvedores ágeis achem que o projeto de arquitetura tradicional leva à documentação desnecessária, a maioria deles concorda que é importante estar atento à arquitetura quando o sistema em desenvolvimento é complexo.
- Nesses casos, é preciso integrar novas práticas de projeto arquitetural aos modelos de processo ágeis.
- Para tomar decisões arquiteturais no início e evitar reformulações e problemas de qualidade na arquitetura escolhida, os desenvolvedores ágeis devem **antecipar a estrutura dos elementos arquiteturais** com base nas **histórias de usuário**, tão logo elas começarem a ser escritas.

*Criando um protótipo que evidencie a arquitetura de forma explícita aos envolvidos chave, uma equipe ágil pode satisfazer a necessidade de um projeto arquitetural.*

*l.bertholdo@ifsp.edu.br*

# Agilidade e Arquitetura

- Em uma abordagem alternativa, envolvendo Scrum e XP, o arquiteto fornece **histórias de usuário arquiteturais** e trabalha com o ***product owner*** para priorizá-las junto às **histórias de usuário de negócio**, à medida que as ***sprints*** são planejadas.
- Inicialmente, o arquiteto trabalha com a equipe durante a *sprint* para garantir que o sistema terá boa qualidade arquitetural. Se a qualidade se mantiver satisfatória, a equipe segue o desenvolvimento sozinha.
- Uma vez concluída a *sprint*, o arquiteto examina a qualidade do protótipo funcional, antes que a equipe o apresente para os envolvidos em uma revisão de *sprint* formal.

*Projetos ágeis bem executados exigem a **entrega iterativa de artefatos** (incluindo a documentação de arquitetura) a cada sprint. Examinar esses artefatos, à medida que evoluem a cada sprint, é uma forma útil de garantir a qualidade da arquitetura.*

*l.bertholdo@ifsp.edu.br*

# Referências


- ENGHOLM JÚNIOR, Hélio. **Análise e Design:** orientado a objetos. São Paulo: Novatec Editora, 2013.

- PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de software:** uma abordagem profissional. 8. ed. Porto Alegre: AMGH, 2016.

- SOMMERVILLE, Ian. **Engenharia de software**. 9. ed. São Paulo: Pearson Prentice Hall, 2011.