O capítulo 7, *Arquitetura*, inicia com uma introdução ao conceito de arquitetura de software, na qual, acredita que a mesma se preocupa com um "projeto em mais alto nível". Ou seja, o foco deixa de ser a organização e interfaces de classes individuais e passa a ser em unidades de maior tamanho. Além disso, a introdução do capítulo fala que os componentes não possuem apenas um "maior tamanho", mas também, devem ser relevantes para que um sistema atenda seus objetivos. Ademais, existe ainda uma segunda definição para arquitetura de software, do qual, acredita que arquitetura de software inclui as decisões de projeto mais importantes em um sistema. Portanto, essa segunda forma de definir arquitetura é mais ampla do que a primeira, ela considera que arquitetura não é apenas um conjunto de módulos, mas um conjunto de decisões.

Padrões arquiteturais propõem uma organização de mais alto nível para sistemas de software, incluindo seus principais módulos e as relações entre eles, sendo eles:

- **Arquitetura em camadas:** É um dos padrões arquiteturais mais usados. Em sistemas que seguem esse padrão, as classes são organizadas em módulos de maior tamanho, chamados de camadas. As camadas são dispostas de forma hierárquica.
- **Arquitetura em Três Camadas:** Esse tipo de arquitetura é comum na construção de sistemas de informação corporativos. As três camadas dessa arquitetura são as seguintes:
    - Interface com o Usuário - responsável por toda interação com o usuário.
    - Lógica de Negócio - implementa as regras de negócio do sistema.
    - Banco de Dados - armazena os dados manipulados pelo sistema
- **Arquitetura MVC:** Conhecido como Model-View-Controller, é utilizado em conceitos de orientação a objetos. MVC define que as classes de um sistema devem ser organizadas em três grupos:
    - Visão - classes responsáveis pela apresentação da interface gráfica do sistema
    - Controladoras - classes que tratam e interpretam eventos gerados por dispositivos de entrada
    - Modelo - classes que armazenam os dados manipulados pela aplicação e que têm a ver com o domínio do sistema em construção.
- **Microsserviços:** Nesse modelo, grupos de módulos são executados como processos independentes, reduzindo o risco de efeitos colaterais causados por mudanças em componentes específicos. As principais vantagens dos microsserviços incluem a evolução rápida e independente de cada componente, permitindo que diferentes equipes trabalhem

simultaneamente sem interferências. No entanto, essa arquitetura apresenta algumas desvantagens significativas. A comunicação entre módulos distribuídos aumenta a complexidade do sistema, já que é necessário dominar protocolos específicos, como HTTP/REST.

- **Arquiteturas Orientadas a Mensagens:** Essa arquitetura busca a comunicação entre clientes e servidores através de um terceiro serviço que tem como a única função prover uma fila de mensagens, dessa forma os clientes, atuando como produtores de informações, inserem mensagens na fila e os servidores, consumidores de mensagens, retiram as mensagens da fila e processam a informação. Com o uso de filas de mensagens, a comunicação pelo lado do cliente torna-se assíncrona, com isso, as filas de mensagens viabilizam duas formas de desacoplamento entre os componentes de uma aplicação distribuída:
    - Desacoplamento no espaço - clientes não precisam conhecer os servidores e vice-versa.
    - Desacoplamento no tempo - clientes e servidores não precisam estar simultaneamente disponíveis para se comunicarem.
- **Arquiteturas publish/subscribe:** Nessa arquitetura as mensagens são chamadas de eventos. Os componentes da arquitetura são chamados de publicadores (publishers) e assinantes (subscribers) de eventos. Em Arquiteturas publish/subscribe oferecem desacoplamento no espaço e no tempo. No entanto, existem diferenças entre publish/subscribe e sistemas baseados em filas de mensagens:
    - No modelo publish/subscribe, um evento notifica todos os seus assinantes, criando uma comunicação de 1 para n (em grupo). Já nas filas de mensagens, cada mensagem é consumida por um único servidor, configurando uma comunicação 1 para 1 (ponto-a-ponto).
    - No publish/subscribe, os assinantes recebem notificações assíncronas quando o evento ocorre. Nas filas de mensagens, os servidores precisam buscar (pull) as mensagens diretamente da fila.
- **Outros Padrões Arquiteturais:**
    - Pipes e Filtros: É um tipo de arquitetura orientada a dados, onde programas, conhecidos como filtros, processam as informações recebidas na entrada e

produzem uma nova saída. Esses filtros são interligados por pipes, que funcionam como buffers.

- Cliente/Servidor: Esse tipo de arquitetura é amplamente utilizado na implementação de serviços básicos de rede. Nela, há dois módulos principais: clientes e servidores, que se comunicam através de uma rede. Os clientes enviam solicitações ao servidor e aguardam a resposta após o processamento.
- Arquiteturas peer-to-peer: Esse tipo de arquitetura distribuída permite que os módulos da aplicação atuem tanto como clientes quanto como servidores. Ou seja, esses módulos, conhecidos como pares, desempenham simultaneamente o papel de consumidores e provedores de recursos.

Por fim, o capítulo termina falando sobre anti-padrões arquiteturais, na qual, uma organização de sistemas não é recomendada. O anti-padrão mais conhecido é o big ball of mud, responsável por descrever sistemas nos quais qualquer módulo comunica-se com praticamente qualquer outro módulo, não possuindo uma arquitetura definida, ao invés, é uma alta no número de dependências.